ISSN 0956-9979 JULY 1996

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **Tradition holds sway.** Once again, *Virus Bulletin* has laboured long and hard; the result of all this graft is the latest DOS scanner comparative review. Turn to p.12 for the details.

• **A rat by any other name…** Stainless Steel Rat is a complex virus which uses various new tricks to fool the unsuspecting user. Learn about these on p.8.

• **Integrity rules.** Wolfgang Stiller, developer of the well-known *Integrity Master*, is still working hard at making his product the best at what it does. Read a little about the man and his cockatiel on p.6.

CONTENTS

EDITORIAL

Merely an Extension of the Concept...

In the course of doing the tests for a comparative review, there is usually one issue that manages to raise itself above all others and to catch my attention. Last month it was the problem with Michelangelo; this month, it is those with Concept. The initial problem with detecting the Concept virus is this: given a *Word* document, how can we tell whether or not it is infected? The attempts to find a solution follow a predictable course. The first thing an anti-virus vendor would do on receipt of the virus would be to release a search-string for it, which would search the whole file for the string. If it is found, the file has the virus. Simple.

“ the ‘perfect’ solution is to open every file … and check it for macro viruses if necessary ”

This approach has several problems, however. First, it takes much longer to search the whole file for the presence of a virus than it does to search only the relevant parts. Second, it is prone to false alarms – when a macro is deleted from a *Word* template, the macro data is still present within the file; it has merely been ‘unlinked’. This is analogous to the removal of a file under most operating systems – the data is not destroyed, just unlinked. It will be overwritten at some point in the future, whenever the system happens to use that area of the disk again. Third, it is possible (if an infrequent occurrence) for *Word* to split the macros. Such files could evade this type of detection.

After the simple search-strings came an intermediate solution – products themselves written in *Word* macros, designed to use the functionality of *Word* itself to delete the macros (this is considerably easier than modifying the file without the help of *Word*). These have disadvantages, too: they don’t catch all files loaded into *Word*, only those brought in by certain routes; and they can be very slow.

Scanners represent the current state of the art, because they understand the format of OLE files (the file type used by *Word*). They can check only the relevant areas of the file for the presence of a virus. This solution offers faster scan times and fewer false positives.

Now that the problem of finding macro viruses within a file is largely solved (for some companies, at least), another problem suddenly crops up. In which files do we look for these viruses? With more traditional viruses, there is no problem: products have an ‘extension list’, which describes which files it should be checking; for example, all products scan .EXE and .COM files; most scan .SYS, .BAT, .DLL, some scan .SCR, .386… it soon mounts up.

However, *Word* documents can be held in files of any extension. The *de facto* standard is to use the extension .DOC for *Word* documents, but it is not necessary to stick to this. Indeed, the *Macintosh* version of *Word* does not use file extensions, *Macintoshes* having a much more elegant solution to the problem of identifying the type of a file, so if *Word* documents are shared between *Macs* and *PCs*, it is quite likely that they will not have the expected extension. Even if a file’s extension is .DOC, this does not prove that it is a *Word* document; I have several packages (*Norton Utilities* and *Telix*) which come with ASCII files with .DOC extensions.

What to do? Scan just .DOT? Just .DOC? Both .DOT and .DOC? Everything? The most widely used technique is to scan .DOT and .DOC, with the recommendation that if a macro virus is found, all files should be checked. Unfortunately, this results in the scanner becoming quite considerably slower – on a given PC, the scanner is checking many more files than before.

The ‘perfect’ solution is to open every file, establish whether or not it is an OLE file, and check it for macro viruses if necessary. Alas, this will result in an even worse deterioration in performance – a couple of test programs which I wrote illustrate the point: one simply listing all the files on my PC took two seconds to run, whereas another which determines whether or not each is an OLE file took 40 seconds. This does not mean that a product which opts for this technique will become twenty times slower, of course, but there will inevitably be an enormous slowdown.

So, beware – when you clean up a Concept infection, make sure you haven’t only cleared it from those files which your anti-virus product has elected to check. You could be in for a nasty surprise when the infection recurs because someone is using non-standard extensions for their *Word* files.

NEWS

The Word is Wazzu

Despite the fact that it first put in an appearance several months ago, the macro virus called Wazzu is the cause of several infection reports received by *Virus Bulletin* in recent days. This virus was posted in source form to the Internet newsgroup alt.comp.virus in early April 1996, and has since been seen in several infection incidents.

Wazzu is different from most other *Word* viruses because it manages to contain all of its functionality in one macro, predictably called AutoOpen. Most viruses require multiple macros to allow the virus to behave correctly, whether it is being run from the Global template (NORMAL.DOT – if this is the case, the *Word* setup on this machine is already infected) or if it is a new infection. Wazzu instead uses a more efficient technique; that of checking the name of the host document.

The virus' payload is ingenious, but simple – every time the AutoOpen macro is invoked (i.e. every time a document is opened in *Word*), Wazzu picks a random number greater than zero and less than one. If this number is less than 0.2 (a one in five chance), the virus will move a word from one point in the document to another. It does this three times, so there is a slightly less than 50% chance (48.8%, to be exact) that at least one word will be moved. After these three tests, it picks a final random number, again between zero and one: if this is less than 0.25 (a one in four chance), the virus will insert the word 'wazzu', followed by a space, at a random point in the document.

There is little else interesting about this particular virus; however, once it is detected on a system, it is important to realize that all documents on that PC which have been accessed since the virus' initial infection must be checked to ensure that they have not been modified by the virus. Each document has a 61.6% chance of being modified in some way, every time it is opened. This checking must be done by eye to ensure that two words have not been swapped. Naturally, the find function can be used to search for the word 'wazzu'.

The original posting to Usenet came from a seemingly well-meaning individual who claimed that the virus had 'become active where I work'. Unfortunately, posting the virus in source form to Usenet simply made it widely available to anyone, so not only is it now in the wild, but we can also expect variants to be written.

The name 'wazzu' is apparently used, especially in the north-western United States, to refer to Washington State University (based in Pullman, Washington) – it is impossible to say whether or not the virus hails from there. If you believe you are at risk, contact your anti-virus vendor for a product update ▮

Prevalence Table – May 1996

Virus	Type	Incidents	Reports
Concept	Macro	55	14.2%
Parity_Boot	Boot	36	9.3%
Form	Boot	33	8.5%
AntiEXE	Boot	32	8.2%
AntiCMOS.A	Boot	24	6.2%
NYB	Boot	20	5.2%
Ripper	Boot	15	3.9%
Empire.Monkey.B	Boot	14	3.6%
Junkie	Multi	12	3.1%
EXEBug	Boot	11	2.8%
Telefonica	Multi	11	2.8%
Quandary	Boot	8	2.1%
Stealth_Boot.C	Boot	8	2.1%
Stoned.Angelina	Boot	8	2.1%
Sampo	Boot	7	1.8%
Jumper.B	Boot	6	1.5%
Natas.4744	Multi	6	1.5%
V-Sign	Boot	6	1.5%
AntiCMOS.B	Boot	5	1.3%
Tentacle	File	5	1.3%
Azusa	Boot	4	1.0%
J&M	Boot	4	1.0%
She_Has	Boot	4	1.0%
WelcomB	Boot	4	1.0%
Boot.437	Boot	3	0.8%
Empire.Monkey.A	Boot	3	0.8%
Manzon	File	3	0.8%
Stoned.Manitoba	Boot	3	0.8%
Tequila	Multi	3	0.8%
Bye	Boot	2	0.5%
Crazy_Boot	Boot	2	0.5%
Stoned.Standard	Boot	2	0.5%
Swiss_Boot	Boot	2	0.5%
TaiPan.438	File	2	0.5%
Other[1]		25	6.4%
Total		388	100%

[1] The Prevalence Table also includes one report of each of the following viruses: Burglar.1150, Cascade.1701, Chance.B, Da'Boys, Diablo, DiskFiller, Fairz, Green_Caterpillar, HDKiller, Helloween, Jackal, Keybug.1704, Mabuhay, Neuroquila, One_Half.3544, Patras.1472, Revenge.1127, Rhubarb, RPS, Russian_Flag, Stoned.NoInt, Stoned.Spirit, Unashamed, Urkel, USSR.492.

IBM PC VIRUSES (UPDATE)

The following is a list of updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 21 June 1996. Each entry consists of the virus name, its aliases (if any) and the virus type. This is followed by a short description (if available) and a 24-byte hexadecimal search pattern to detect the presence of the virus with a disk utility or a dedicated scanner which contains a user-updatable pattern library.

Type Codes

- **C** Infects COM files
- **D** Infects DOS Boot Sector (logical sector 0 on disk)
- **E** Infects EXE files
- **L** Link virus
- **M** Infects Master Boot Sector (Track 0, Head 0, Sector 1)
- **N** Not memory-resident
- **P** Companion virus
- **R** Memory-resident after infection

NOTE: The template given in January's IBM PC Virus Update Table for the virus Annihilator.596 was inaccurate; the correct template reads as follows:

```
Annihilator.596     60E8 0000 582D 8B01 958D B6AC 01E8 0200 EB13 B917 018B FEBA
```

Babol.2048 **CR:** An appending, 2048-byte virus containing the plain-text strings: '*.exe' and '(c) Dj_Babol'. The payload, which triggers on 13 June, includes corrupting the contents of the hard disk.

```
Babol.2048          B8FF FFCD 213D AAAA 7414 33C0 8EC0 CD12 B106 D3E0 2639 064A
```

Euskara.811 **CN:** An appending, 811-byte direct infector containing encrypted text (in Basque), which it displays: 'Milaka Urtez Eutsi Dugu Eta Milaka Urtez Eutsiko. Euskara, Jalgi Hadi Plazara.' It hooks Int 09h.

```
Euskara.811         0545 2680 3D45 7510 FCF3 A48C C08E D8B4 25B0 09BA 00FA CD21
```

Geek.734 **CENP:** A prepending (COM) and companion (EXE) 734-byte direct infector containing the text: 'Out of memory', 'Get lost, geek' and 'GONEPATH ;\dos ;'.

```
Geek.734            89E5 33FF 8E06 2C00 32C0 B9FF 7FF2 AEAE E0FB 83C7 02B8 003D
```

IOE.155 **CN:** An overwriting, 155-byte direct infector. It infects two files at a time and contains the text: 'Internal OPCode error.$*.com'.

```
IOE.155             B440 B99B 0181 E900 01BA 0001 CD21 7204 B43E CD21 B409 BA7D
```

IOE.239.B **CN:** An overwriting, 239-byte direct infector which contains the text: 'Devil (C)1996 by FT c:\DOS' and 'Internal opcode error.$*.com'. The virus can be detected using the template for IOE.239 [*VB, May 1996*].

Iota.56 **CN:** An encrypted overwriting, 56-byte direct infector containing the text: '*IOTA*.COM'.

```
Iota.56             B91F 00BE 0801 8034 ??46 E2FA B138 CD21 C3
```

IVP.1067 **CRN:** An encrypted, appending, 1067-byte fast infector containing the text: 'ZOMBIE.incXcracow Znalic ZOMBIE.PROD MADE THIS VIRII AS A DEDICATION TO Abdul Alhazred – The Old ones were, The Old ones are,and The Old ones shall be – Copyright (c) 1996 Xcracow Znalic for "THE NEW ZOMBIE.PROD" [IVP]'.

```
IVP.1067            8D9E 1401 B905 042E 8A27 2E32 A62F 052E 8827 43E2 F2C3 ??
```

Jasio.666 **CR:** An appending, 666-byte virus which contains the encrypted text: 'JASIO' and '*.COM'. All infected files have their time stamp set to 22 seconds. The virus hooks interrupts Int 21h and Int 09h, but the infection procedure does not activate before the Int 21h service routine has been called 30,000 times.

```
Jasio.666           B903 00B4 40CD FFB9 9402 33D2 B440 CDFF 595A B801 57CD FFB4
```

Kerstin.923 **EN:** An appending, 923-byte, fast direct infector. It contains the encrypted text: 'Kerstin.95b' and 'Happy birthday Kerstin ! I'll always be there 4 U. Contact me. In love A.'

```
Kerstin.923         B440 0E1F 8BD6 83EA 1FB9 9B03 CD21 26FE 068D 0026 8E1E AC00
```

Linc.196 **CR:** An appending, 196-byte virus which resides in the Interrupt Vector Table. It contains the string 'Winter'. Infected files have their time stamp set to 62 seconds.

```
Linc.196            A359 02B4 40BA 2002 B9C4 00CD 21B8 0042 9931 C9CD 21B4 40B9
```

Linc.228 **CR:** An encrypted, appending, 228-byte virus residing in the Interrupt Vector Table. It contains the string 'Autumn'. Infected files have their time stamp set to 62 seconds.

```
Linc.228 (files)    50EB 38BE ???? 5B4B 4B87 378D 7810 E8C8 00??
Linc.228 (memory)   B440 B903 00BA 5802 CD21 595A 80C9 1FB8 0157 CD21 B43E CD21
```

Linc.307 **CR:** An appending, 307-byte virus containing the text: ''The Waxwork Crew' proudly release their first virus 'aardvark''. Infected files have their time stamp set to 2 seconds.

```
Linc.307            C33D 7698 7504 B8FF FFCF 3D00 4B75 7B50 53B4 43CD 2172 71B8
```

Linc.318 **CR:** An encrypted, prepending, 318-byte virus containing the text: '[Sleeping]'. Infected files start with the word B805h.

```
Linc.318            B93E 01BA 3E02 C3E8 0600 9C2E FF1E 1802 BDEE FE2E 80B6 1802
```

LosLobos.535

CN: An appending, 535-byte, fast, direct infector containing the plain-text strings: '????????COM', '*.666', and 'bailos los lobos'. Infected files start with the byte 0E8h (near call).

```
LosLobos.535        EB04 B44C CD21 8DB6 0301 BF00 01FC A5A4 B02A 8AE0 CD21 3C00
```

MrS.323

CR: An appending, 323-byte virus containing the text: 'SCHUBERT 1797-1828.'. The virus intercepts the procedure writing to a file (Int 21h, function 40h) and may write the above message instead. Infected files have the string: 'Mr' located at offset 0003h.

```
MrS.323             813E 6704 CEDE 7451 A113 04C7 0667 04CE DE48 A313 0492 B106
```

PMM.575

CR: An appending, 575-byte direct infector, infecting up to three files at a time. The virus contains the text: '*PMM r1.0*', which is located at the end of infected files.

```
PMM.575             0EE8 0000 5F81 C754 011F AC3C 7073 6C3C 4073 F7A8 C474 0924
```

Polish.1769

CER: An encrypted, appending, 1769-byte virus. It avoids infecting programs which match the following patterns: *SC*.*, *OS*.*, *V*.*, *H*.* and *I*.*. Infected files are n*16+7 bytes long.

```
Polish.1769         9090 0E1F BE?? 00B9 D306 8CC3 BF?? 000E 07FC AC34 ??AA E2FA
```

Poppins.256

EN: A companion virus, about 256 bytes long, which contains the plain-text string: '<+PoPPinS+>' and the encrypted text: 'Bad command or file name'.

```
Poppins.256         BEC8 01B9 1A00 AC48 92B4 06CD 21E2 F7BA C501 B824 2501 06E4
```

Reverse.948

CER: An appending, 956-byte virus containing the encrypted text: 'Reverse-948 i 1. Created by Renata G. from Lubin City in Sept 1993 moc.dnammocexe.niamcn'. The virus infects COMMAND.COM by overwriting the end of file (usually with zeros) and avoids infecting NCMAIN.EXE (*Norton Commander*).

```
Reverse.948         2EFF 063B 03B8 BADC CD21 3DCD AB74 528C D848 8ED8 8B1E 0300
```

Shire.117

CN: An appending, 117-byte direct infector, infecting one file at a time. It contains the text: '*.com' and '+Time+'. Infected files are n*2+1 bytes long.

```
Shire.117           B44E 8D54 65CD 2172 10D0 0EB0 FFB4 4F72 F4BA B4FF B802 3DCD
```

Shire.143

CN: A prepending, 143-byte direct infector containing the string: '*.?Om'. Infected files have their time stamp set to 62 seconds.

```
Shire.143           B44E BA89 012B C9CD 2172 4CA0 AEFF 40B4 4F24 1F74 F0BA B6FF
```

Shire.149

CN: An appending, 149-byte direct infector, containing the text: '*.COM' and '*LAVA*'. Infected files are n*2+1 bytes long.

```
Shire.149           B44E 8D94 8300 CD21 7210 D00E B2FF B44F 72F4 BAB6 FFB8 023D
```

Shire.155

CN: An encrypted, prepending, 155-byte direct infector, containing the text: '*.CoM' and 'MrTiny' (the last three characters are left unencrypted). Infected files are n*2+1 bytes long.

```
Shire.155           BE0A 01B9 6C00 8034 ??46 E2FA C3BA 08FF B99B 00CD 21B8 0042
```

Shire.199

CN: An appending, 199-byte direct infector infecting three files at a time. It contains the text: ':<Your Sinclair>:'. Infected files are n*2+1 bytes long.

```
Shire.199           E812 00B8 034E 8D95 4300 8885 4200 CD21 7318 BA80 00B4 1AB9
```

Shire.210

CN: An encrypted, appending, 210-byte direct infector containing the text: '-Kiss-' and '*.cOm'. Infected files have their time stamp set to 62 seconds.

```
Shire.210           B9AE 0080 760F ??45 E2F9 C3B0 03CF E9??
```

Shire.220

CN: An encrypted, appending, 220-byte direct infector containing the text: '-Purple-' and '*.cOm'. Infected files have their time stamp set to 62 seconds.

```
Shire.220           B9B8 0080 760F ??45 E2F9 C3B0 03CF E9??
```

Shire.253

CN: A slightly polymorphic, appending, 253-byte direct infector containing the text: '*.com', '+TIME+' and '+Chemical Clock+'. Infected files are n*2+1 bytes long.

```
Shire.253           CD21 7210 D00E B0FF B44F 72F4 BAB4 FFB8 023D CD21 5F72 2693
```

Shire.300

CN: An encrypted, appending, 300-byte direct infector containing the text: '*EFIL' and '*.COM'. Infected files have their time stamp set to 62 seconds.

```
Shire.300           BE?? ??BB 0001 8777 01BF 6CFE 5703 F3B9 9600 F3A5 C3E8 0201
```

SillyC

CN: Two SillyC variants, 105 and 107 bytes long. Both are appending, fast, direct infectors which contain the string: '*.COM'. Infected files are n*2+1 bytes long.

```
SillyC.105          E821 00CD 2189 044E B169 E813 0048 48CD 21B1 03E8 0A00 B43E
SillyC.107          E822 00CD 2189 4401 B16B E814 0048 48CD 21B1 03E8 0B00 B43E
```

Sirius.361

CN: An encrypted, appending, 361-byte virus which infects programs on drive C only. It contains the text '<1994>' and '*.COM'. Infected files have their time stamp set to 6 seconds.

```
Sirius.361          8D76 19E8 0200 EB10 8A96 6301 B94A 018B FEAC 32C2 AAE2 FAC3
```

StayCool.573

CR: A stealth, appending, 573-byte virus containing the text: 'Louise Broderick my princess Written at Barclays plc Softare Labs Stay Cool Mickey Athwel'. Infected files have their time stamp set to 2 seconds.

```
StayCool.573        B830 30CD 2181 FB30 3074 F20E 1FB4 4ABB FFFF CD21 B44A 83EB
```

Trivial.45.K

CN: An overwriting, 45-byte, fast, direct infector containing the string: '*.COM'.

```
Trivial.45.K        BA9E 00CD 2172 0FB7 40B9 2D00 BA00 0193 CD21 B43E CD21 B44F
```

INSIGHT

A Master of Integrity

Wolfgang Stiller, owner of *Stiller Research* and a cockatiel called Sport, is the chief developer of *Integrity Master*. Born in 1955 to German parents in Wilhelmshaven, West Germany, Stiller and his family emigrated west, arriving in the US in 1959. He grew up in a typical American family, with brother Rick and sister Evelyn, in Michigan and Ohio.

The Wonder Years

Programming was not Stiller's first career choice: it was only during his Chemistry and Physics studies at Michigan State University that he began in earnest, on a 4K *DEC PDP 8L*. This he used to collate data from experiments, and soon found it more fun than his studies. A dual major in Chemistry and Computer Science led to specialising in operating systems. By the time he finished his studies at Florida State University, he was working full-time as a programmer on *CDC Cyber*-series mainframes.

During those *CDC* days, Stiller wrote a mini operating system for the *Intel* 8008 microprocessor S-100 bus micro, which he burned into a 2K EPROM: 'It wasn't much,' he admits, 'but it was all mine, it fitted in 2K of EPROM, and let me use my micro as a semi-smart mainframe terminal.'

Later, he bought a CPM-based micro with 64K of memory and dual floppies: 'Boy, did I feel extravagant!' he laughed. A corrupted executable on this system led to Stiller becoming aware of the problems involved in data integrity, and later, in viruses: 'When I reloaded the program file from a backup,' he recalled, 'it was fine. Obviously the program had been damaged somehow. It troubled me that something could damage my program without me being aware of it.'

To combat the problem, Stiller wrote a rudimentary CRC program – a basic integrity checker which allowed him to ensure that his programs were not corrupted. Years later, he realised such an approach would be a valid anti-viral tactic.

The *CDC Cyber* series led in 1973 to *IBM* mainframes, on which Stiller developed Florida's on-line Worker's Compensation system. This complete, he switched back to operating systems, and worked on the *IBM MVS*.

A Problem within a Problem

In 1988/89, Stiller updated his *PCdata* toolkit, which he had written in the late 1970s in Z80 machine code. This he adapted for viruses: the finished product consisted of a set of assembly language programs and an article published in *PC Magazine* (US) – with unexpected results: 'The article,' explained Stiller, 'described the type of changes a virus would make, and how to use these programs to detect and remove viruses. The program checked files and system sectors for changes; this provided a no-cost solution which anyone could use to check for viruses.

'I asked users to send me reports of what they discovered. This provided me with virus samples and a good idea of the limitations of this approach. More importantly, I discovered that viruses were rarely the cause of data corruption. For each virus report, I received over 100 reports of otherwise undetected data corruption from causes other than viruses.'

Developing Integrity

Stiller Research has been in existence since 1988. Its original focus was OS development, but specialisation in security and data integrity led to the production of Integrity Master.

This, says Stiller, will now continue to be the company's focus: 'We're the only company with a product that provides complete data integrity protection as well as virus scanning and generic virus detection in one integrated product. Our product is a 100% highly-optimized assembler. I'm not aware of any other product that can provide the speed and comprehensive integrity checking of our product.'

At the moment, Stiller is searching for a partner able to apply more 'marketing muscle' to promoting the product, admitting readily that finances do not allow an advertisement campaign to match activities of companies like *Symantec* and *McAfee*: 'As we're OS specialists,' he added, 'we will probably be offering *Win95/NT* enhancement and diagnostic utilities. These are spin-offs from the work we are doing with viruses and *Integrity Master*.'

He anticipates a time when he can devote himself 100% to his product's technical development: 'I am committed to keeping a solid integrity-based product on the market, but I may be more effective as chief technical officer, with someone else at the helm.'

If Stiller has a 'pet peeve', it is that people seem to view viruses as the only threat to the integrity of programs and of data: 'Most people these days have some type of virus protection, but only few have any way of assuring the integrity of their data from other threats. This is equivalent to having a health insurance policy against only one disease.

'System conflicts, software bugs, hardware failures; all account for more damage than viruses, yet for most people, they may go undetected. It's vital for anyone using a PC for business to verify the integrity of their programs and data!'

Flying into Danger

Polymorphic viruses, asserted Stiller, will continue to appear, but will grow less rapidly than has hitherto been the case: 'Polymorphics have become old-hat to the virus

Stiller with constant companion, Sport: 'I'm not so sure he really understands 80x86 assembler…'

writers, and most anti-virus producers can now add polymorphic detection fairly rapidly (it has also become old-hat for us).'

Stiller views the biggest new threat as macro viruses, believing their numbers will grow rapidly over the next two years. His rationale is that a user with few programming skills ('Which describes most virus writers, by the way!') can quickly write a new macro virus, or modify an existing one to evade scanners.

'We of course will see macro viruses written for products other than *Word*, *Excel*, and *Ami Pro*. Any product that supports a flexible macro language will probably be host to such viruses. Other products will be a lesser problem: most are not as rich in features that can be exploited by a virus (e.g. overriding menu options, autoexec macros), and their files are not exchanged to the extent of *Word* documents.'

Stiller also feels that heuristics, albeit an aid in the battle, have severe limitations: 'I don't believe they are reliable enough for use by many non-technical users,' he said. 'A technical user would not be alarmed if a heuristic program announced that a disk utility did direct disk access and used undocumented functions, but a less technical user might see that the anti-virus program "said they had a virus" rather than understanding the heuristics.'

Integrity Master's use of heuristics is limited, in conjunction with other information from the integrity checking component. Stiller is only too aware that customer overreaction to a warning from the product can sometimes be more destructive than a virus attack: 'We have to be very careful how we warn our customers about suspicious changes.'

Stiller sets himself very high technological standards, and professes himself distressed by the current trend for huge, slow, buggy software: 'We now use high-level language code for *Windows*-specific areas but even with *Windows*, we can get huge performance gains by using assembler in core areas of our product. I get tremendous satisfaction from developing software that meets my stringent standards. Next to my family this is the most satisfying thing in my life.'

Now You See Them, Now You Don't...

Stiller sees the recent subsumation of so many smaller companies into giant multi-nationals as unfortunate: 'Frequently we see good products (e.g. Untouchable) just disappear. It's to everyone's advantage to have numerous options, innovation, and healthy competition. This will disappear if a small number of companies dominate the market. In my opinion, we are in great danger of this now.

'We lose many sales because some companies refuse to buy anti-virus technology from a smaller, less known company such as ourselves. This is how *MSAV/CPAV/Norton* have dominated the US market. I take the word "integrity" very seriously: we don't allow our agents to make questionable claims. But some customers select products based on these claims (e.g. "detects all known and unknown viruses").'

It is no longer possible, according to Stiller, to release a brand-new anti-virus product with virus-specific recognition, because with the amount of known viruses in existence, the work involved would simply be too much. However, he foresees the possibility of the smaller side of the market being dominated by new generic products, or larger products licensing their scanning technology to other companies.

On Cockatiels and Cessnas

Both Stiller and his wife Dianne have pilot's licences: he is instrument-rated, and she has a commercial licence (though her job is as CICS OS programmer for *MCI*). They make innumerable trips in their *Cessna 172* with their cockatiel, Sport: 'Sport enjoys looking down on other birds from the plane,' added Stiller. 'I think he likes being "top bird". When we land in instrument conditions, Sport keeps an eye on things. I'm not certain, but he seems to understand how the glide slope and localizer needles work (an abstract concept).'

As to family, Stiller is satisfied with the two closest to him: 'Dianne and Sport take all my time. We are Sport's "flock" (at least, that's how he see us) so we feel it's important to include him in as much as possible. Some people think we are crazy (the way we dote on Sport) but Sport gives us much more than we give him. God has blessed me with both Sport and Dianne.'

Sport has the run of home and office, and is Stiller's constant companion: 'When things get really rough and I get frustrated, he preens my hair and peps me up. He has occasionally found a problem on a listing, but I'm not so sure he really understands 80x86 assembler…'

Leisure time is taken up with Sport and Dianne: when not flying, they can be found skiing in Colorado, bicycling in Europe or the US, canoeing, caving, rafting, or hiking. Nonetheless, Stiller describes himself as 'a low risk-taker who does everything to eliminate unnecessary risks'.

Understatement and high standards seem to be core concepts in Stiller's life: whatever he does, bears this stamp, and will doubtless continue to do so.

VIRUS ANALYSIS 1

Stainless Steel Electronic Rat

Eugene Kaspersky

It is almost as if there is a competition being played out in the virus underworld: who can write the largest and most complex virus? No sooner do I put Zhengxi and Nutcracker in the box labelled 'Done' than the next one is knocking on my electronic door. Once again I am like Duke Nukem at the last level, face to face with a new techno-monster with a new battery of tricks, interrupts in his hands, looking down at me, waiting for my faults, and I have just a keyboard…

Our new guest is very complex. Infected files grow by exactly 18364 bytes, but actual virus code is about 11K of assembler (the length of its TSR code is the total length minus that of the polymorphic decryption loops). The virus uses a number of polymorphic, anti-debugging, and stealth tricks, and so has a tendency to halt the system from time to time: in some cases it corrupts files it is trying to infect.

Installation

The virus is encrypted with three polymorphic engines, thus there are three decryption loops – each loop decrypts the next, until the third decrypts down to the raw code. During decryption, the decryptors use anti-debugging tricks, and numerous junk commands, subroutines and branches.

When the virus code receives control, SSR performs another decryption loop. The majority of the virus code is encrypted with a more complex internal encryption routine (thus the bulk of the code hides behind four levels of encryption).

This final loop is different from the others – instead of simply looping across the body of the code and decrypting, it installs an Int 01h (Single Step) handler, and then executes a dummy loop. It does not decrypt the code directly, but on each instruction, the virus' Int 01h handler is called, and this decrypts the next byte of the virus. Thus decryption takes place, but in a fairly indirect fashion.

During decryption, SSR does not use constant commands for each byte, but encrypts each with a different operator, taking those required from a list (SUB, ADD, XOR, ROL).

SSR now passes control to the installation routine, which first checks for the presence of DEADh at address 198h in segment 0 – if this is found, installation aborts. During installation, SSR hooks Ints 01h, 13h, and 2Fh; traces Int 2Fh to get its original address; gets the original Int 13h address through the undocumented Int 2Fh call (AH=13h); then steals a block of system memory by patching the MCB list, and copies itself there.

To hook Int 21h, the virus gets and saves two bytes from the start of the current Int 21h handler, overwrites (patches) these with a call to Int ACh and then hooks Int ACh. In this way it receives control whenever an Int 21h is issued, but without directly hooking it. SSR also hooks Int 1Ch (timer), Int ABh – it uses the latter to call to the infection routine.

To help conceal its code in memory, the virus erases the memory area occupied by its code whilst it was going resident – as the TSR copy is encrypted, there is no decrypted virus code in memory. Then SSR restores the host program's code, and returns control to it.

Int 21h Handler

When the patched Int 21h handler passes control to the virus handler, SSR restores the original Int 21h handler (replacing the two bytes overwritten earlier), and gets the number of the function called.

Five DOS functions are intercepted: 4B00h (Load and Execute), 43h (Get/Set File Attributes), 3Dh (Open Existing), and 4Eh/4Fh (FindFirst/Next by Name). On file access calls, the virus calls its infection routine, and on FindFirst/Next calls, it decreases the length of infected files (stealth routine).

When one of these calls is made, the virus decrypts its main infection and stealth routine, calls it by issuing an Int ABh, re-encrypts it, and returns control to the original Int 21h handler. To re-patch the Int 21h code, when the instruction flow passes that code, the virus temporarily hooks Int 2Ah. When that handler receives control (i.e. an Int 2Ah has been issued), the virus patches the original Int 21h handler and unhooks from Int 2Ah.

The virus intercepts several other Int 21h functions – these are all installation checks for different viruses and anti-virus software. When one of these is caught, it checks the filename of the calling program (see 'Trigger Routines').

Infection

When a file is accessed, SSR calls its infection routine. First, it checks the file extension – it infects only COM and EXE files. Then it opens the file, checking its date and time stamp to prevent duplicate infection (see below).

It reads the file header, checks the internal file format, and encrypts itself with its internal encryption routine and polymorphic engines. The resulting encrypted code is saved to the end of the file.

SSR then changes the file's entry point, by modifying the EXE file header, or patching the COM file header with a JMP VIRUS instruction. Next, it saves and overwrites data at offset 0040h in COM files, and at offset 200h from the EXE header in EXE files. I see no reason for corruption of this nature other than to prevent disinfection of files by CRC. Then the virus closes the file, and exits the infection routine.

The virus also tries to process COM files which begin with JMP NEAR or CALL NEAR instructions by writing the JMP VIRUS to the destination address of the initial JMP CALL: this fails, and the instruction is written to the beginning of the file as for other files.

Features

SSR pays special attention to *PKLITE*-compressed files. It writes its encrypted code to the end of the file using the method described above, but writes the JMP VIRUS instruction into the middle of the file; into the routine *PKLITE* uses when decompression is going to fail due to lack of memory. The virus then patches the standard *PKLITE* unpacking routine so control passes to this routine regardless of how much memory is available. Thus the virus receives control.

During infection, the virus hooks Int 24h to prevent the standard 'Write protect error' message. It also temporarily unhooks Int 13h, and gets and restores the file's attributes and date/time stamp.

When SSR opens a file for the purpose of infection, it uses another unusual trick to fool anti-virus monitors and to hide its own activity. Before it opens a file, it temporarily hooks Int 2Ah, saves and sets the first byte of the filename extension to zero, and performs an Open for Read/Write call. The virus does not then issue a call to open the file 'Filename.Ext', but 'Filename.0xt' (where '0' represents the zero byte): anti-virus monitors will pay no attention to such calls.

"the virus can also detect attempts to trace the Int 21h handler code … and [will] halt the PC"

When DOS receives this call, it performs an undocumented Int 2Ah call from its kernel. This is intercepted by the virus' handler – it restores the filename, releases Int 2Ah, and returns control to the DOS kernel, and DOS opens Filename.Ext without an Open for Read/Write being issued for that file.

The virus determines whether a file is already infected by examining the time and date stamp. On infection, SSR modifies this stamp such that the value of the seconds field is equal to twice the value of the month field (e.g. 22.05.96 12:00:10). Files so marked are not infected.

Trigger Routines

There are several trigger routines in the virus code. SSR's Int 21h handler checks for the following DOS functions:

- FFh: FLU-SHOT anti-virus monitor; installation check of the Jerusalem.Sunday, Tumen, and Hero viruses
- ABCDh: PME.Burglar virus installation check
- 4B53h: Horse and One_Half viruses installation check
- CCCCh: no idea
- DEADh: installation check of several different viruses

If the virus detects any of these functions, it calls the trigger routine: it plays a sound effect (on a 386 this is a siren) and displays the following message:

```
    !!! ALARM WARNING DANGER APPROACHING !!!
Hacker-fucker TSR shit or Any Virus Detected !!!
    !!! ALARM WARNING DANGER APPROACHING !!!
Hacker-fucker TSR shit or Any Virus Detected !!!
Anyone who wants to fuck Revenge is Naivnij Man
     With best wishes & thanks to DialogScn
Emulation engine will have problems with this ZHOM
In future versions we will add :
 1. Protected Mode Decryptor [VMME]
 2. Adinf table Hacker-cracker
 3. Destroy Files/Disks/CMOS/Printer/CDROM
 4. Disk encryption and other BUGs,GLUKs & SHITs!
Dis is only BEGIN… Win95 & her lamers must die!
           Searching… SEEK & DESTROY
            There can be only one …
```

When the infection routine closes the file, SSR checks its internal counter, which is incremented on each Int 1Ch call. Depending on the counter, the virus may call the second trigger routine, which displays a message ('This is Revenge of Stainless Steel Rat'), and waits for the escape key to be pressed. It then clears the screen, overwrites a random sector of drive C, and halts the PC. Also, the display may shake.

The virus can also detect attempts to trace the Int 21h handler code, in which case it will display a message in Russian, corrupt the CMOS checksum field, and halt the PC.

While infecting a file, the virus checks the filename – it does not infect files which match the following patterns: DR*.* (DRWEB), AI*.* (AIDSTEST), AD*.* (ADINF), HI*.* (HIEW), CO*.* (COMMAND.COM), AV*.* (AVP), WI*.* (WIN), KE*.* (?), US*.* (?), GD*.* (?). When a file with the string ?ID*.* in its name (AIDSTEST) is executed, SSR displays a message in Russian, and halts the computer.

The virus also checks the extension of the file being searched by FindFirst/Next DOS functions, and deletes files with the extensions PAR, PIF, ICO, WEB, PAS, BAS, AVB, and FRQ. It then forces the Find function to return the string 'shit !' in Cyrillic coding, instead of the filename.

Location	Text-string
Start of virus code	Hi Hacker! Welcome to Hell
Installation routine	Move over, I said move over Hey, hey, hey, clear the way There's no escape from my authority- I tell you
Infection routine	Gimme the prize, just gimme the prize
Code which writes encrypted virus code to files	Save me,save me
In EXE header processing code	Don't lose your header
FindFirst/Next handler	I'm the invisible man
File deletion routine	Now you DiE !
Routine for processing PKLITEd files	Crazy Little Thing Called PkLite

Figure 1: SSR has various text-strings, which are contained in different routines throughout the virus, as shown above.

The virus also contains the strings:

```
Give me your WEBs, let me squeeze them in my hands,
Your puny scaners,
Your so-called heuristics analyzers,
I'll eat them whole before I'm done,
The battle's fought and the game is won,
I am the one the only one,
I am the god of kingdom come,
- THERE CAN BE ONLY ONE -
I'M GOING SLIGHTLY MAD
Just very slightly mad !
All dead...
THIS IS OF STAINLESS STEEL RAT
Revenge virus v 1.02 released at 22.04.96
Copyright (c) 1996-97 2 Rats Techno Soft
Written by Stainless Steel Rat
StealthedMetamorphicCrazyForcedSynthesatedRandom
MegaLayerEncryptionProgressionMutationEngineGenerator
S.S.R.
BUGS INSIDE <tm>
```

The polymorphic engines used by the virus contains the texts:

```
RandomEncryptionSynthezator ■ S.S.R. 1996-97
```

```
THE STAINLESS STEEL RAT MUTATION ENGINE v 1.21 beta
  ■■(c) S.S.R. 1996-97 ■■
```

```
Metamorphic Mutation Engine v 2.00 (C)
Stainless Steel Rat 1996-97
It's C00LEST Engine
```

[*Editor's note: There is much here for those interested in the source for text strings in viruses – SSR's strings are drawn from the film 'Highlander' and the music of Queen (both that which featured in the film and that which did not). Perhaps the more obscure source is reserved for what the virus author calls his virus – the Stainless Steel Rat is a character in the books of Sci-Fi author Harry Harrison.*]

SSR

Aliases:	MME.Ssr.
Type:	Memory-resident parasitic COM and EXE infector, polymorphic, encrypted in system memory.
Self-recognition in Memory:	Word at 0000:0198h is set to DEADh.
Self-recognition in Files:	Seconds field of time stamp equal to month multiplied by two.
Hex Pattern in Memory:	5859 5E9D E825 01E8 4801 CDAB E8EC 00E8 6B01 EB22 90E8 6501 E8E0 009C 0650 532E (No hex pattern is possible in files)
Intercepts:	Int 21h for infection, Int 1Ch (timer) for trigger routine.
Trigger:	See text.
Removal:	Under clean system conditions identify and replace infected files.

VIRUS ANALYSIS 2

Putting the Boot In

Kevin Powis

Boot.437 is a virus that has been successful in the wild for quite some time. As the name suggests, it is a boot sector virus – when an infected floppy is in the drive during the boot process, unless the PC has been instructed via the BIOS not to boot from diskettes, the virus will be loaded into memory at segment zero, offset 7C00h, and control passes to that address from the firmware in the standard fashion.

Boot.437 first preserves the original Int 13h handler by reading it from offset 4Ch in segment 0 and preserving it for later use. It then installs itself into memory, by deducting 1KB from the memory size word at offset 413h, segment 0. There is what appears to be an attempt to break up the (very generic) sequence of instructions used to do this – one superfluous operation is present. This is, of course, no defence against a scanner which knows Boot.437 specifically, it only helps against some generic systems.

The next stage is to calculate a segment register corresponding to the now missing 1KB of conventional memory. This is obtained by the *de facto* standard code snippet which takes the reduced memory word mentioned above and shifts it left by 6 bits. The result is a new segment register value allowing the virus to access its new home in memory. When I see these code snippets turn up byte-for-byte in numerous viruses, I wonder if the original authors wish they could claim copyright or royalties on these useful modules…

With the memory reserved and a segment pointer obtained for that memory, Boot.437 copies its own image to the new location and passes control to the next instruction, but in the copy image. The original image in segment 0 is now redundant.

On receiving control, Boot.437's new image issues a disk reset command via the standard BIOS call, usually reserved for when a disk error has been detected before attempting a retry. However, Boot.437's author seems to be trying to pre-empt any such problem.

The virus is now installed, and must locate and execute the original boot sector stored on infection. At this point, 437 does not know if it is running courtesy of an infected hard disk or whether someone has booted from an infected floppy. This is determined by examining the byte at offset C3h within the virus body – this contains the physical number of the target drive stored when the disk was infected. If this is 80h or greater, the virus is deemed to have executed from a hard disk, otherwise it was from a floppy.

If this byte indicates the virus is being executed from a hard disk, the virus' work is almost complete. If, however, it indicates a floppy disk, Boot.437 calls the infection routine (see the next section). When control returns from this, it

reads in the original sector (which the infection routine saves), hooks Int 13h by replacing the vector with a pointer to its own code, and passes control to the original sector.

Infection Sub-routine

This routine carries out all infections for the virus. First, the target disk's boot sector is read into memory: if the target is a hard disk, the partition table is examined to determine the location of the active partition. If the partition table does not contain an element marked as active, the disk is spared infection. If an active partition entry is found, the virus reads the boot sector of that partition into memory.

The infection routine now compares six bytes at offset 100h in the virus body with the bytes at the same position in the sector being examined – this is an infection test. If they match, the disk is already infected, and infection aborts. Otherwise, the sector is secreted for future reference, and the image in memory is patched with three virus bytes at the start: these represent a jump to the bulk of the virus code (437 bytes, hence the name) which is placed into the sector at offset 3Eh. The modified sector is then written back to disk, completing the process.

Interrupt Handler

Once the Int 13h handler installed at the end of the installation routine is in place, all disk access requests will pass through the virus code. The virus is only interested in read and write requests – all others pass through unhindered.

In the case of a hard disk access request, a test is done to compare the target drive against the one from which the virus was loaded (using the value at C3h). If they match, the target is already infected and is left alone; the original disk request being allowed with no attempts at stealth. This allows the virus to infect quite happily any number of PC hard disks.

Boot Sector Infection

The Master Boot Sector, located at head 0, cylinder 0, sector 1, is the first sector on any PC hard disk. When created and partitioned with the standard DOS FDISK program, the MBS is divided into three areas: bootstrap (or loader) code, partition table and boot sector signature (word AA55h at the end of the sector). The partition table starts at offset 01BEh and consists of four sixteen-byte elements: each can describe a partition. On a simple one-hard-disk system, one partition on the disk, containing the operating system loader, will be tagged as the active partition. The first sector of this partition, commonly called the DOS (or Partition) Boot Sector, contains the first part of the OS loader.

Most boot sector viruses infect the MBS on hard disks by overwriting the loader with their own code and using part of the remainder of track 0 as storage space for the virus body. Some, however, infect the DOS Boot Sector of the active partition. This is more difficult to do correctly, as the virus must parse the partition table to locate the start of the active partition, then load in the first sector of that partition, which it can then modify. Boot.437 does this, but the most widespread example of this type of virus is Form.

Floppy disk accesses are treated differently. Boot.437 examines another low memory word which indicates (amongst other things) which floppy disk drive motors are currently spinning. The virus masks the unwanted bits, then checks to see if a floppy drive is currently active. The floppy will only be infected if the motor on that drive is spinning; a ploy to eliminate signs of spurious disk activity [*Other viruses also use this trick; e.g. Jumper. See VB, April 1995, p.11. Ed.*].

In all cases, if the disk is considered suitable for infection, Boot.437 invokes the infection routine described above. Control then passes to the original Int 13h handler.

Storage Algorithm and Partition Tables

Where the virus places the copy of the clean boot sector depends on the type of disk being infected; in head/cylinder/sector notation, the positions are as follows: 360K 1.0.3, 720K 1.0.5 (both of these place it at the end of the root directory), 1.2MB 1.0.13 (placing it over the penultimate sector of the root directory), and 1.44MB 1.0.17 (placing it over the the second cluster in the data area – part of the first file on the diskette will be overwritten).

Summary

As shown by the WildList, Boot.437 is a successful virus. I would imagine every scanner in existence can detect it [*In the comparative review on p.12, every scanner except MSAV could. Ed.*], yet it is still in the wild. Its code, whilst effective, demonstrates inefficiencies which may indicate a learning curve on the part of the author. The virus carries neither trigger nor payload, and is content simply to infect and to replicate.

Boot.437

Aliases:	None.
Type:	Floppy boot sector infector; fixed disk active partition infector.
Infection:	Potentially, all floppy and fixed disks used in a host system.
Self-recognition:	Six bytes at offset 100h in sector compared with virus code.
Hex Pattern:	This pattern will locate the virus on hard and floppy disks and in memory. `80FA 8072 083A 16C3 0074 1FEB 1A50 558B ECC7 4602`
Intercepts:	Interrupt 13h disk handler.
Trigger:	None.
Payload:	None.
Removal:	Hard disk – FDISK /MBR. Diskette – salvage any required files (which will be unaffected by the virus), then format.

COMPARATIVE REVIEW

Scanning the Skies

Six months have passed already since *Virus Bulletin* last did a comparison of DOS scanners, which can only mean one thing – it's time for another look.

This time, 23 scanners were submitted for testing. They were pitted against the usual four test-sets, which have been updated since the last review in January 1996. There have been some changes of technique since then, which are also described below.

Test-sets

As in last month's *Windows 95* comparative, the Boot Sector set now consists solely of samples taken from the *WildList*. Its full name, therefore, is the In the Wild Boot Sector set. This allows an 'Overall In the Wild' percentage to be calculated, combining the In the Wild file and boot scores.

The deadline for products to be submitted to *Virus Bulletin* for inclusion in the review was mid-April 1996, so the *WildList* used is that of the previous month, March 1996.

The Standard test-set has been revamped, with many new viruses drawn from the last year. Also, the naming of those viruses already in the set has been updated.

The Polymorphic test-set has undergone its usual expansion: it now comprises 10,000 samples of 20 viruses. Readers will note that seven of these twenty are drawn from the *WildList*. The remaining thirteen are library viruses which were chosen for their polymorphism.

Technique

There has only really been one change in the comparative technique since the last DOS review; however, it is a fairly significant one. Where previously the scanner would be run once per test-set, or once per virus group, *Virus Bulletin* is now running the scanner once per virus sample.

For the Boot Sector test samples, a complete image of an infected diskette is written to a disk in the drive, which is then scanned: this process eliminates most of the vast amount of disk swapping otherwise required. Were a scanner to miss any sample in this way, it would later be given the original sample disk, in case there was a problem with the image; however, no problems of this type were encountered.

Such a process entails quite an overhead (running the scanner 10,798 times takes considerably longer than running it four times, even if the same number of samples are scanned), but it does prevent products adjusting their sensitivity dependent on what has gone before.

Of course, it was not necessary to execute each of 23 scanners over ten thousand times by hand: this would clearly be at best very boring, and at worst intractable. The entire process was automated with an elaborate series of batch files, and the results recorded for later parsing.

Detection testing was performed on a network of eight DOS clients connected to a single *NetWare* server. The DOS clients were 386s or 486s, with varying specifications: this allowed eight products to be tested at once, each running from, reading virus samples from, and writing log files to, the server. The different specifications of these machines are not significant, as they were not used for speed tests.

All speed tests were performed on the same machine under the same configuration (see Technical Details). Scanner speed was measured against a clean floppy (43 COM/EXE files totalling 997,023 bytes), an infected floppy (the same files infected with Natas.4744; 1,201,015 bytes), and a clean hard disk (3250 COM/EXE files; 196,338,487 bytes over 65 directories). This last doubles as a false positive test: any false positives encountered are mentioned in the text.

Calculations

The only change in the calculations for this review is the method by which the 'Overall In the Wild' percentage is obtained: instead of combining In the Wild File and Boot percentages directly, the two are regarded as a single set, and the percentage recalculated. This prevents omissions in the Boot Sector set (which is smaller than the File set) having an disproportionate effect on the result. Readers are encouraged to consult the last few comparatives [*VB, July 1995 p.14; January 1996 p.13*] and the testing protocols [*VB, February 1995 p.12; November 1995 p.14*].

There is also a detailed document describing the exact calculation system, with worked examples, available on the *Virus Bulletin* World Wide Web site (see Technical Details for a URL).

Readers will also notice the absence of the 'Overall' scores used in previous reviews. This figure was misleading in several ways, and the very name encouraged people to view that single figure as the review's be-all and end-all – it was used to justify statements such as 'the product placed third in the *Virus Bulletin* comparative'. Rather than attempt to make such statements more accurate by applying weighting factors to the test-set percentages before combining them into the 'Overall' result, the figure has simply been removed.

Extra Tests

In addition to detection and speed tests, the scanners are run against other tests: detection in memory and disinfection. The boot sector viruses AntiCMOS.A, AntiEXE.A,

Empire.Monkey.B, and Form.A, and the file viruses Burglar and Manzon, were used for these tests.

These particular viruses were chosen as they are in the wild, and affecting real users; they should therefore be dealt with correctly by a good anti-virus product. Disinfection of the boot sector viruses is tested both on hard and on floppy disks.

It would be useful to be able to test the products more thoroughly in this area; however, the nature of the problem means that it is considerably more difficult (and also takes commensurately longer) to carry out the tests in these categories than to do those in the simple scanner results and speed sections. All of these tests were done on a selection of old *Amstrad* portables which had only 1MB of memory. This fact caused problems for some products.

	Clean Floppy		Infected Floppy		Clean Hard Drive	
	Scan Time (min:sec)	Data Rate (KB/s)	Scan Time (min:sec)	Data Rate (KB/s)	Scan Time (min:sec)	Data Rate (KB/s)
Alwil AVAST!	0:50	19.5	1:23	14.1	1:39	1936.7
Cheyenne InocuLAN	0:44	22.1	0:53	30.1	4:50	661.2
Command F-Prot	0:40	24.3	0:53	22.1	0:55	1743.1
CSE PCVP	0:27	36.1	31	37.8	0:56	3423.9
Cybec VET	0:52	18.7	0:56	20.9	0:55	3486.1
DialogueScience DrWeb	1:33	10.5	1:48	10.9	37:03	86.3
EliaShim ViruSafe	0:43	22.6	0:38	30.9	1:18	2458.2
ESaSS ThunderBYTE	0:28	34.8	0:33	35.5	0:35	5478.2
H+BEDV AVScan	0:58	16.8	1:05	18	2:32	1261.4
IBM AntiVirus	0:58	16.8	1:09	17	5:17	604.8
Iris AntiVirus Plus	0:42	23.2	0:55	21.3	4:25	723.5
KAMI AVP	1:18	12.5	0:57	20.6	16:20	195.6
Leprechaun Virus Buster	0:33	29.5	0:48	24.4	1:13	2626.5
Look Software Virus ALERT	0:54	18	1:29	13.2	1:44	1843.6
McAfee Scan	0:37	26.3	0:38	30.9	3:07	1025.3
Microsoft Anti-Virus	0:40	24.3	0:50	23.5	1:44	1843.6
Norman Virus Control	0:57	17.1	1:00	19.5	2:24	1331.5
RG Software Vi-Spy	0:51	19.1	0:56	20.9	1:38	1956.5
Sophos SWEEP	0:48	20.3	0:34	34.5	3:03	1047.7
Stiller Integrity Master	1:04	15.2	3:40	5.3	2:42	1183.6
Symantec CPAV	0:50	19.5	0:50	23.5	Incomplete	Incomplete
Symantec Norton AntiVirus	0:54	18	0:46	25.5	1:11	2700.5
S&S Dr Solomon's AVTK	0:51	19.1	1:02	18.9	1:24	2282.6
Trend PC-cillin	0:48	20.3	0:42	27.9	3:26	930.8

Alwil AVAST! v7.50.05

ItW Boot	98.7%	Standard	98.6%
ItW File	99.8%	Polymorphic	97.3%
ItW Overall	99.3%		

In *VB's* January DOS comparative review, *AVAST!* suffered a slight setback to its usual high-scoring ways: this trend now appears to have been reversed, a fact to which the percentages bear witness. Performance in the In the Wild test-set was dented only by missing Crazy_Boot and one of the four Concept samples (of which more later). That it missed only three samples from the Polymorphic test-set gives it a very good score – it places third in this category.

As mentioned above, the single In the Wild File sample which was missed was one of Concept. It is curious that, if this sample is scanned along with several others, the virus will be correctly detected as Concept. If it is scanned on its own (as in this test), or placed in a directory before the other samples, it will be pronounced clean. As there appears to be no logical reason for this, it must be assumed to be a bug.

The product does not offer any file repair, and hard disk boot sectors can be fixed only if a recovery diskette is available for the machine concerned. If a virus is found in a floppy disk boot sector, BGUARD (a program from the *AVAST!* suite) can overwrite the sector. All of the viruses were found in memory.

Cheyenne InocuLAN v4.0, 3.15a

ItW Boot	98.7%	Standard	87.6%
ItW File	98.0%	Polymorphic	62.1%
ItW Overall	98.3%		

	ItW Boot		ItW File		ItW Overall	Standard		Polymorphic	
	Number	Percent	Number	Percent	Percent	Number	Percent	Number	Percent
Alwil AVAST!	76	98.7%	311	99.8%	99.3%	402	98.6%	9997	97.3%
Cheyenne InocuLAN	76	98.7%	304	98.0%	98.3%	336	87.6%	6205	62.1%
Command F-Prot	77	100.0%	312	100.0%	100.0%	371	94.1%	5456	51.2%
CSE PCVP	53	68.8%	216	71.0%	70.1%	268	77.6%	4192	41.0%
Cybec VET	77	100.0%	307	98.0%	98.9%	391	97.3%	8520	81.6%
DialogueScience DrWeb	57	74.0%	298	96.4%	86.7%	401	98.2%	9766	95.5%
EliaShim ViruSafe	76	98.7%	296	95.9%	97.1%	321	85.3%	4647	42.3%
ESaSS ThunderBYTE	77	100.0%	312	100.0%	100.0%	398	98.0%	8471	82.6%
H+BEDV AVScan	68	88.3%	275	91.5%	90.1%	348	91.0%	6583	62.6%
IBM AntiVirus	77	100.0%	311	99.5%	99.7%	399	98.2%	7389	65.2%
Iris AntiVirus Plus	76	98.7%	307	99.0%	98.9%	393	98.0%	9354	89.6%
KAMI AVP	72	93.5%	308	99.0%	96.6%	408	99.8%	10000	100.0%
Leprechaun Virus Buster	55	71.4%	217	67.8%	69.4%	316	82.4%	4737	46.6%
Look Software Virus ALERT	76	98.7%	307	99.0%	98.9%	405	99.5%	8502	82.9%
McAfee Scan	77	100.0%	307	99.0%	99.4%	362	91.9%	6446	65.2%
Microsoft Anti-Virus	12	15.6%	98	33.0%	25.4%	176	58.9%	975	9.0%
Norman Virus Control	67	87.0%	305	98.5%	93.5%	397	98.2%	9998	98.7%
RG Software Vi-Spy	75	97.4%	292	94.6%	95.8%	367	93.3%	5662	50.1%
Sophos SWEEP	77	100.0%	301	97.3%	98.4%	401	98.6%	9498	93.4%
Stiller Integrity Master	72	93.5%	299	96.8%	95.4%	397	97.3%	5268	46.9%
Symantec CPAV	66	85.7%	298	95.0%	91.0%	282	79.9%	4886	46.5%
Symantec Norton AntiVirus	72	93.5%	307	99.0%	96.6%	363	92.8%	5747	59.7%
S&S Dr Solomon's AVTK	77	100.0%	307	99.0%	99.4%	405	99.5%	9487	90.7%
Trend PC-cillin	75	97.4%	303	98.0%	97.7%	302	84.2%	8751	81.1%

InocuLAN is definitely improving: its ItW score is respectable now, and shows a distinct upturn from a year ago. However, the ItW detection of the other products is similarly improving: everybody must simply work that much harder to keep up. The file viruses missed were Burglar and Ph33r.1332; the boot sector virus, Chance.B. In the other sets, things are also on the up, but perhaps not as dramatically: *Cheyenne* appears, rightly, to be concentrating on In the Wild viruses above all else.

On the other tests, the table shows that this product cured three of the four boot sector viruses in memory (with the exception of AntiCMOS.A), and removed all of the boot sector viruses correctly from hard and floppy disks. Both of the file viruses were missed in memory, and only Manzon was disinfected (bytes were left behind, but the files ran without problems).

Command Software F-Prot v2.22

ItW Boot	100.0%	Standard	94.1%
ItW File	100.0%	Polymorphic	51.2%
ItW Overall	100.0%		

Finally, a product which gets 100% on the Overall ItW score: this is great to see, as in the last few comparatives no one managed this. It is perhaps fitting that the first product to be certified under the current *NCSA* system is also the first for some time to have an equivalent performance in a *VB* test. However, in the Polymorphic set, the product places an undistinguished seventeenth.

All boot sector viruses were detected in memory and disinfected on disk; the file viruses were missed in memory, but disinfected correctly from files.

	Memory				Hard Disk				Floppy Disk				Memory		File	
	AC	AE	E	F	AC	AE	E	F	AC	AE	E	F	B	M	B	M
Alwil AVAST!	F	F	F	F	*	*	*	*	D	D	D	D	F	F	F	F
Cheyenne InocuLAN	F	D	D	D	D	D	D	D	D	D	D	D			F	D+
Command F-Prot	F	F	F	F	D	D	D	D	D	D	D	D			D*	F
CSE PCVP	F	F		F	F	D	F	D	F	D	F	D				
Cybec VET	F	D	D	F	D	D	D	D	D	D	D	D		F	D*	F
DialogueScience DrWeb	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D*	D
EliaShim ViruSafe	F	F	F	F	D	D	D	D	D	D	D	D		F	F	D
ESaSS ThunderBYTE	F	F	F	F	D	D	D	D	D	D	D	D			D**	D
H+BEDV AVScan	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F
IBM AntiVirus	F	F	F	F	D	D	D	D	D	D	D	D	F	F	F	F
Iris AntiVirus Plus	D	D	D	D	D	D	D	D	D	D	D	D	D		D*	D+
KAMI AVP	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D*	D
Leprechaun Virus Buster	F	F	F	F	D		D	D	D	D	D	D				
Look Software Virus ALERT	F	F	F	F	*	*	*	*	D	D	D	D	F	F	F	F
McAfee Scan	F	F	F	F	D	D	D	D	D	D	D	D	F	F	D*	D+
Microsoft Anti-Virus				F				D				D				
Norman Virus Control		F	F	F	D	D	*	D	D	D	D	D	F	F	F	F
RG Software Vi-Spy		F	F	F	D	D	D	D	D	D	D	D	F	F		
Sophos SWEEP	F	F	F	F	F	D	D	D	F	D	D	D			F	F
Stiller Integrity Master	F	F	F	F	D	D	*	D	D	D	D	D		F	F	F
Symantec CPAV	F	F	F	F	D	D	D	D	D	D	D	D			D*	F
Symantec Norton AntiVirus	F	F	F	F	D	D	D	D	D	D	D	D	F	F	F	D
S&S Dr Solomon's AVTK	F	F	F	F	D	D	D	D	D	D	D	D	F	F	D*	D
Trend PC-cillin	F	F	F	F	D	D	D	D	D	D	D	D			D!	D+

Virus Key: AC = AntiCMOS.A AE = AntiEXE.A B = Burglar E = Empire.Monkey.B F = Form.A M = Manzon

Performance: F = Found D = Disinfected * = Recovery disk required
D* = Byte 12h in header (part of the Checksum word) different from original, files functional.
D+ = Varying numbers of virus bytes left behind, files functional and non-infective.
D** = Bytes 0Eh/0Fh (Initial SS value), 11h (part of Initial SP value), and 12h (part of Checksum word) incorrect, files functional and non-infective.
D! = Repaired incorrectly, files non-functional.

CSE PCVP v2.23

ItW Boot	68.8%	Standard	77.6%
ItW File	71.0%	Polymorphic	41.0%
ItW Overall	70.1%		

This product's history of problems with the Boot Sector set is continued in this comparative, and the Overall In the Wild score is an unimpressive 70.1%; not an adequate defence. Equally, the Polymorphic score is weak, at 41.0%.

Excepting Empire.Monkey.B, the boot sector viruses were detected in memory: the user must run subsidiary programs to remove them. Of the viruses used, disinfectors were present for Form and Empire.Monkey.B. The file viruses were missed in memory (unsurprising, as *PCVP* does not detect the viruses used). The product is, however, very fast.

Cybec VET v9.0

ItW Boot	100.0%	Standard	97.3%
ItW File	98.0%	Polymorphic	81.6%
ItW Overall	98.9%		

Again, a significant improvement since the last outing for *Cybec's VET* back in January: here, only Concept and BackFormat were missed from the In the Wild set. Polymorphic detection has also shown an upturn; however, there is still a problem with incomplete detection of sample groups in this set. Even resolving this issue would result in quite a significant hike in the resulting percentage.

VET is very quick: coming second only to *ThunderBYTE* in terms of clean hard disk data rate is impressive by any standards, and it does not suffer *TBAV's* false positive rate.

This product detected all the boot sector viruses in memory, and cured both AntiEXE.A and Empire.Monkey.B there. All boot sector viruses in memory were correctly removed from hard and floppy disks. Of the two file viruses, only Manzon was found in memory, and only Burglar could be disinfected.

DialogueScience DrWeb v3.11

ItW Boot	74.0%	Standard	98.2%
ItW File	96.4%	Polymorphic	95.5%
ItW Overall	86.7%		

DrWeb comes out poorly in the detection tests: third from bottom in the all-important Overall In the Wild set, and this despite a very good score on the Polymorphics (95.5% places it fourth in this category).

Its main problem is the awful Boot Sector score: missing twenty samples in this set puts a crimp on any chances of a good result. Given the very low number of viruses (1865) about which the product knows, however, it is amazing that it performs as well as it does. More concentration on viruses at large world-wide would serve the product well.

In the extra tests, *DrWeb* was able to detect and remove all the viruses from memory, both types of disk, and files – the only other product to manage this was *AVP*.

Also worthy of note are the results of the speed tests: this scanner was the slowest tested. However, when combined with the bundled integrity checker, the speed is brought up to an acceptable level. The theory here is fairly standard: the scanner is only invoked when a file has not previously been fingerprinted, or when its fingerprint has changed. Alas, the level of heuristics in the product is revealed by the high number of false positives suffered – twelve in all.

EliaShim ViruSafe v6.8

ItW Boot	98.7%	Standard	85.3%
ItW File	95.9%	Polymorphic	42.3%
ItW Overall	97.1%		

Like several other products, *ViruSafe's* In the Wild score has gone up quite considerably in the last six months – this is good news for their users. The viruses which were missed are Concept, 15_Years, and Byway.B. Detection of Natas.4744, Virogen.Pinworm, and One_Half.3544 is incomplete. Polymorphic detection is poor – *ViruSafe* places second from bottom in this category. This set includes viruses of which the product has no knowledge, and several of which its knowledge is incomplete.

In the extra tests, *ViruSafe* detected all boot viruses in memory, and removed them correctly from all host media. Of the file viruses, *ViruSafe* detected Manzon in memory and disinfected it from files, but missed Burglar in memory and could not remove it from files.

ESaSS ThunderBYTE v7.01

ItW Boot	100.0%	Standard	98.0%
ItW File	100.0%	Polymorphic	82.6%
ItW Overall	100.0%		

It's all swings and roundabouts in the anti-virus industry: *ThunderBYTE* was once unbeatable, but in recent months it has suffered a slight dip in detection. Now, however, its detection is right back up there. The In the Wild test-sets are perfect at 100%: *TBAV* and *Command's F-Prot* are the only products to manage that in this comparative. The Polymorphic rate has also improved distinctly.

The speed is undiminished from previous reviews; it is still as remarkable as ever. Unfortunately, because *TBAV* is so heuristic in nature, it suffers from a higher than average false positive rate: in this test, it encountered five.

H+BEDV AVScan v2.65

ItW Boot	88.3%	Standard	91.0%
ItW File	91.5%	Polymorphic	62.6%
ItW Overall	90.1%		

AVScan's In the Wild score has gone up fractionally since the last comparative, which is definitely a good thing. It would be nice to see faster improvement in this area, however. The other scores are slightly down on last time.

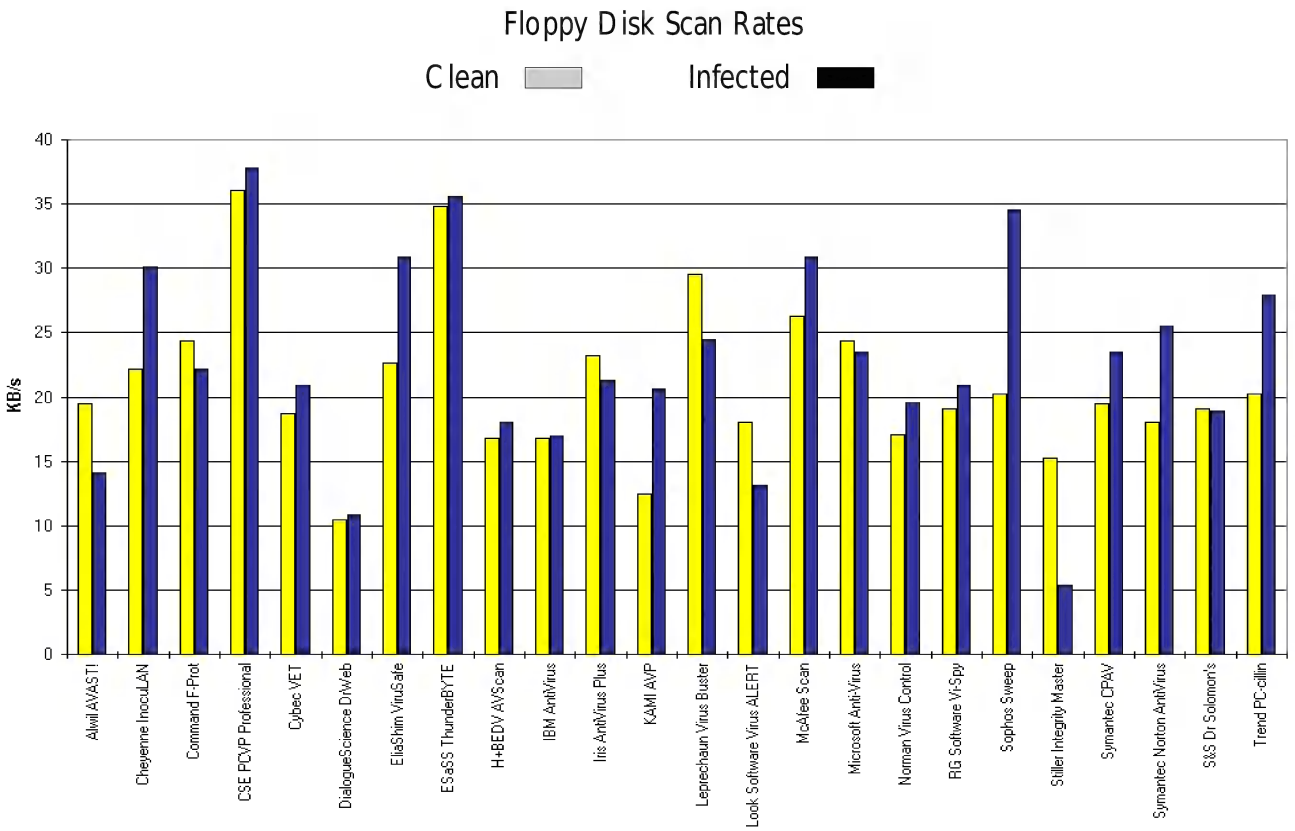
As usual, the shareware version of *AVScan*, which does not offer disinfection, was submitted for review. However, the product did detect all the viruses when they were active in memory. In addition, it suffered two false positives.

IBM AntiVirus v2.4.1B

ItW Boot	100.0%	Standard	98.2%
ItW File	99.5%	Polymorphic	65.2%
ItW Overall	99.7%		

IBMAV is now very close to 100% detection against the ItW test-set – it only misses one of the two No_Frills.Dudley samples, which edges it away from the target, but its performance is very good nonetheless. The score on the Polymorphic set would be much improved if the viruses about which *IBMAV* knows were detected completely. However, this product is still worth a serious look.

There has always been a slight problem testing *IBMAV's* speed on a clean hard drive: the first time the product is run, it checksums all of the executable files. On subsequent runs, the scanner does not check the files for viruses unless the checksum has changed. The hard drive speed figures in the table are those without the checksums: if these are present and no files have changed, the product takes just 38 seconds to check the clean set (a data rate of 5046KB/s), which would place it just behind *ThunderBYTE*.



In the extra tests, *IBMAV* detected all file and boot viruses in memory, and disinfected all boot viruses from hard and floppy disks. It could not remove either Manzon or Burglar from the file samples.

Iris AntiVirus Plus v21.17

ItW Boot	98.7%	Standard	98.0%
ItW File	99.0%	Polymorphic	89.6%
ItW Overall	98.9%		

Almost every review has its dark horse; the product that appears out of nowhere to do much better than its track record indicates that it should. This time around, that product is *Iris AntiVirus Plus*.

Missing only Ph33r.1332 and Chance.B in the In the Wild sets gives it an score of 98.9% – unfortunately, in this day and age this is only sufficient to place it seventh equal (with *Virus ALERT*). However, it is a good score nonetheless. In the Polymorphic set, it gets a very respectable 89.6% – a couple more completely-identified groups will send this score into the 90s with little effort.

In the other tests, the product detected and removed all of the boot sector viruses from memory on both hard and floppy disks. Both file viruses were disinfected on disk – Manzon had varying numbers of bytes left behind. Only Manzon was missed in memory.

KAMI AVP v2.2 (12/04/96)

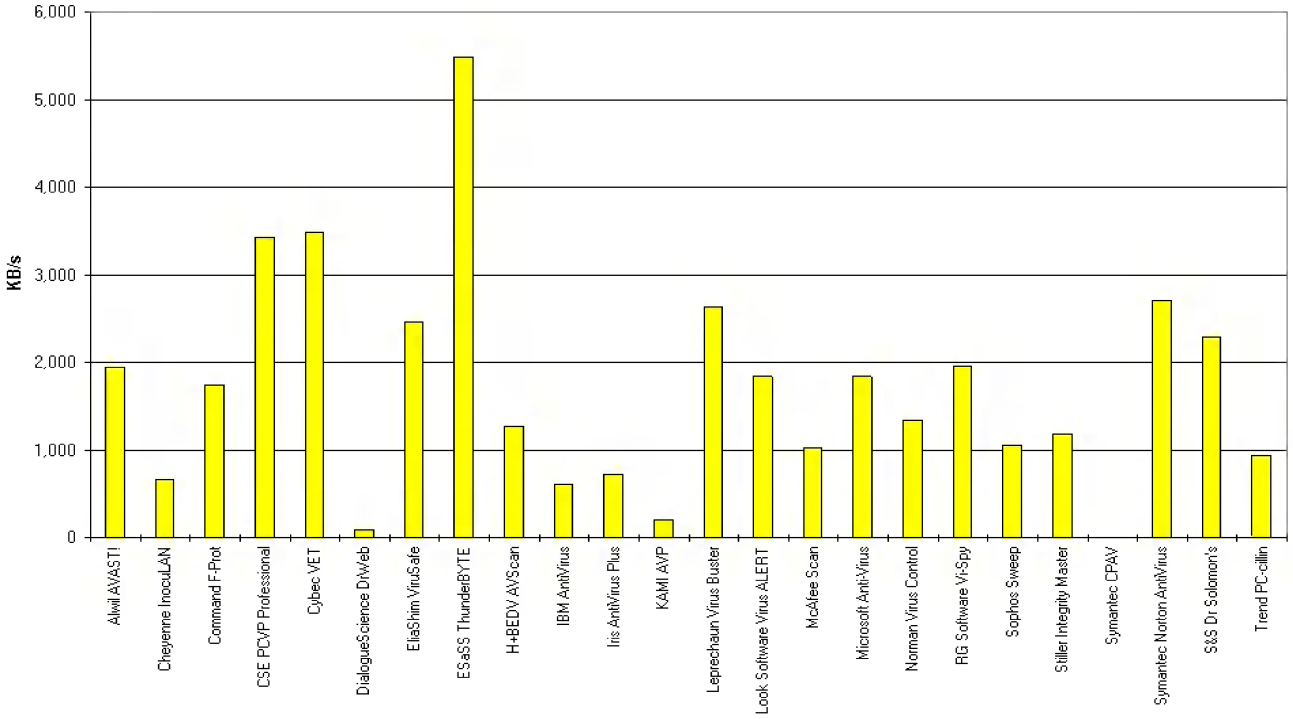
ItW Boot	93.5%	Standard	99.8%
ItW File	99.0%	Polymorphic	100.0%
ItW Overall	96.6%		

It is disappointing that *AVP*, in this release from its US distributor, *Central Command Software*, falls down on the In the Wild test-sets. Missing an overall total of nine in the combined In the Wild grouping is sufficient to drop its score to below average, and is certainly not what is expected from *AVP*. However, something one does expect from this product is high quality polymorphic detection, and that is exactly what is received. With all the virus samples in this set detected, one cannot fail to be impressed.

The price paid for this level of detection (at least of the more obscure viruses) is high, however. The product is very slow: it has a data rate of just under 200KB/s on a clean hard disk, which is simply too slow for everyday use. *AVP* also suffered one false positive.

As far as other tests are concerned, *AVP* cured all the viruses used in memory and disinfected the on-disk copies almost entirely without difficulty. However, it still displays a curious tendency, noted in the last comparative, to try and remove Form from the hard disk's Master Boot Record before successfully removing it from the partition boot sector. Whilst this bug does not threaten any damage, it is still a problem.

Scanning Speeds on the Clean Hard Drive



Leprechaun Virus Buster v4.84.04

ItW Boot	71.4%	Standard	82.4%
ItW File	67.8%	Polymorphic	46.6%
ItW Overall	69.4%		

At this point in time there can be little to recommend this product: 69.4% ItW detection is highly inadequate protection at best. The high point is 82.4% on the Standard set, but this is still nothing to write home about. Against the clean test-set, there were a startling eighteen false positives!

Virus Buster detected all boot sector viruses in memory, and when it came to disinfecting these, it slipped up only on AntiEXE.A, which it could not remove from a hard drive. The file viruses were missed both in memory and on disk.

Look Software Virus ALERT v4.10

ItW Boot	98.7%	Standard	99.5%
ItW File	99.0%	Polymorphic	82.9%
ItW Overall	98.9%		

This Canadian product almost comes up with the goods this time around in the In the Wild set – missing the one sample of Crazy_Boot and the five of Alfons.1344 alone means that it has come a long way since the last comparative. The Polymorphic result shows complete detection of sixteen of the twenty groups, giving the product a score of 82.9%.

Like *AVAST!*, the product can only remove a boot sector infection from a hard drive if a recovery disk is available for the machine in question. It can, however, remove them from floppy diskettes with no problems. No file disinfection is offered, but the file viruses were detected in memory.

McAfee Scan v2.2.11.9603

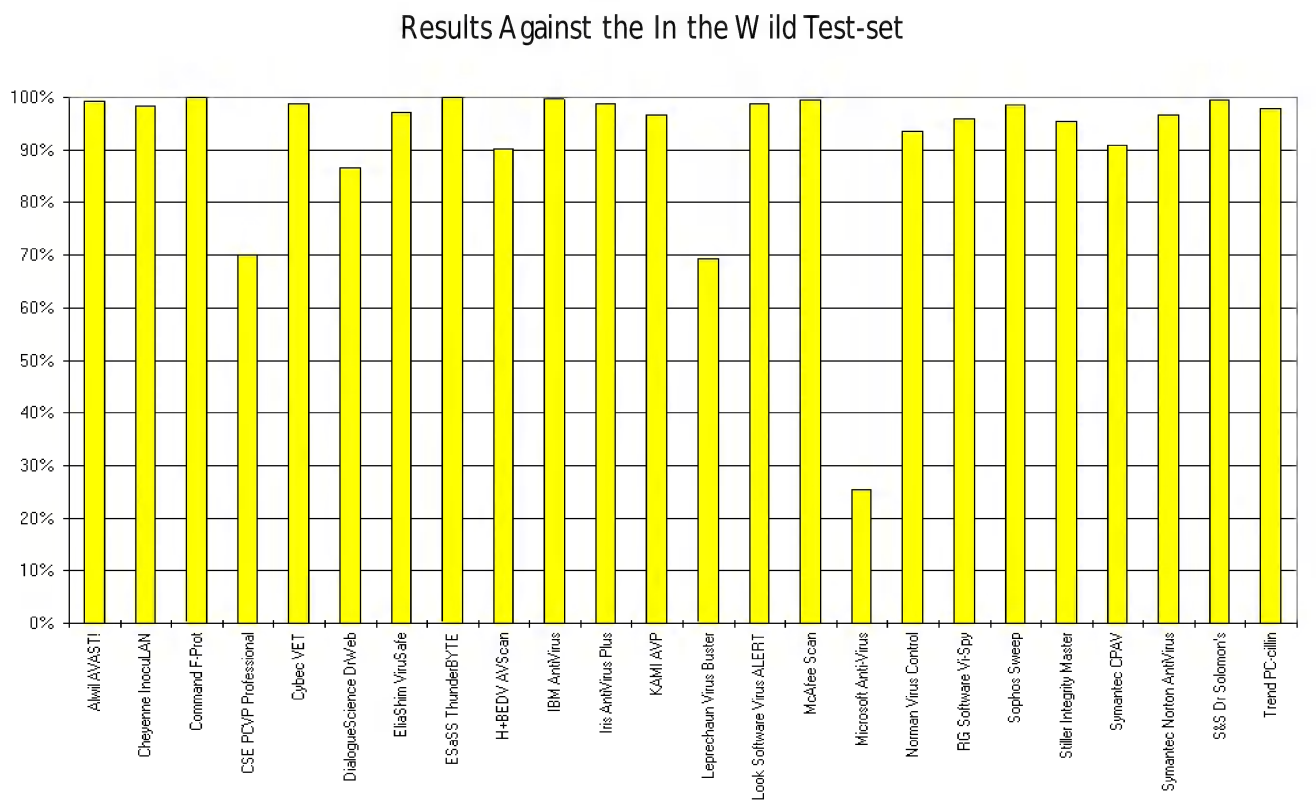
ItW Boot	100.0%	Standard	91.9%
ItW File	99.0%	Polymorphic	65.2%
ItW Overall	99.4%		

Only the five samples of Alfons.1344 stood between this version of *McAfee* and an Overall ItW score of 100%; as it is it drops to 99.4%, which is still highly respectable. The score against the Polymorphic set has dropped slightly since the last DOS comparative: this is due to new viruses in this set. It would be a shame if the improvement seen in *McAfee's* results over the last few comparatives were to slip away…

McAfee found all the boot and file viruses in memory, and also disinfected all the samples (Manzon had extra bytes left behind at the end of the file): a good result on these tests.

Microsoft Anti-Virus v6.22

ItW Boot	15.6%	Standard	58.9%
ItW File	33.0%	Polymorphic	9.0%
ItW Overall	25.4%		



Calls still come from people who use this and believe it must be fabulous because it comes from *Microsoft*... This never was true, and certainly isn't now. Even if you don't want to pay for an anti-virus product, a user can do better than this.

Norman Virus Control v3.51

ItW Boot	87.0%	Standard	98.2%
ItW File	98.5%	Polymorphic	98.7%
ItW Overall	93.5%		

NVC was once a dark horse, but it has again established a good reputation – which is great, apart from the fact that it brings with it high expectations on the part of reviewers.

In the Polymorphic set, these expectations are more than matched: just two samples (both of Code.3952:VICE.05) are missed, giving the product a score of 98.7% in this set – not bad by any standards. In the In the Wild set, however, things are not quite as good: missing ten boot sector viruses and seven file samples (five Alfons.1344 and two Concept) results in a score of 93.5%, which places it eighteenth – a definite 'could do better' on this set.

NVC missed AntiCMOS.A in memory, but did detect the other boot sector and file viruses there. All of the boot sector viruses were disinfected; however, a recovery diskette was needed for Empire.Monkey.B. Neither of the file viruses was disinfected.

RG Software Vi-Spy v14.0.02.96

ItW Boot	97.4%	Standard	93.3%
ItW File	94.6%	Polymorphic	50.1%
ItW Overall	95.8%		

A slight drop in all sets when compared with the last comparative review of six months ago: in the In the Wild sets, *Vi-Spy* missed Chance.B and Crazy_Boot, all five Alfons.1344 and Burglar samples, two of the five samples of Bosnia:TPE.1_4, all three Markt.1533 and Ph33r.1322 samples, and both Manzon samples. *VB* looks forward to these omissions being corrected. In the Polymorphic set, there are a number of incompletely-detected viruses: solving this would up the score quite considerably.

All the viruses (file and boot) bar AntiCMOS.A were found when they were active in memory (the file viruses because of missing memory), and all four of the boot sector viruses were disinfected correctly from hard and floppy disks. Neither file virus was disinfected (perhaps unsurprising, as, unfortunately, *Vi-Spy* did not know about the viruses used). The product encountered five false positives.

Sophos SWEEP v2.84

ItW Boot	100.0%	Standard	98.6%
ItW File	97.3%	Polymorphic	93.4%
ItW Overall	98.4%		

Sophos has a tradition of refusing to be caught out in the *VB* comparatives; however, this time around the In the Wild score is down fractionally from last time. Here it missed Alfons.1344, Burglar, and the samples of Concept with a DOC extension – all easily fixed. Missing these is sufficient to drop the product to tenth place in these sets. The Polymorphic score is well above average, however: it missed only two of Code.3952:VICE.05 and all of Digital.3547.

SWEEP found all boot sector viruses in memory, and disinfected all except AntiCMOS.A from floppy and hard disks. File virus disinfection is not offered, therefore both file viruses were missed in memory.

Stiller Integrity Master v2.61b

ItW Boot	93.5%	Standard	97.3%
ItW File	96.8%	Polymorphic	46.9%
ItW Overall	95.4%		

Integrity Master is a difficult product to review: its CUI menuing interface is different from most around; screens are always packed. Make no mistake, all the information a user could desire is available, but overall appearance is somewhat cluttered. As to detection, more work is required on the In the Wild viruses: 91.0% here is too low for comfort. The score in the Standard set is the high point, but the score on the less-important Polymorphic set is fairly low.

Integrity Master detected all the boot viruses in memory, and removed them all from disks without problems, although to remove Empire.Monkey.B from the hard disk, a recovery disk is required. File virus disinfection is not offered, but Manzon was detected in memory.

As regards the speed tests, one point is worth noting. When scanning an infected diskette, there is a brief pause after each virus is detected. The reviewer could not remove this pause, which has lengthened this particular scan time.

Symantec CPAV 22 March 96

ItW Boot	85.7%	Standard	79.9%
ItW File	95.0%	Polymorphic	46.5%
ItW Overall	91.0%		

Everything which *VB* always says about *Central Point* still holds true: although it has been given a score for the Polymorphic set, it crashed on two of the twenty groups (needless to say, the groups on which it crashed are classed as undetected). In the In the Wild sets, the scores place it firmly towards the bottom of the heap, and the Polymorphic score is unimpressive.

All the boot viruses were found in memory and disinfected from floppy and hard disks; however, both file viruses were missed in memory, and only Burglar could be disinfected. The question must be asked, yet again, why *Symantec* doesn't either simply move all its *CPAV* customers over to *NAV*, or do some more work on *CPAV's* engine, to bring it up to speed.

Symantec Norton AntiVirus (30/04/96)

ItW Boot	93.5%	Standard	92.8%
ItW File	99.0%	Polymorphic	59.7%
ItW Overall	96.6%		

NAV's detection had, in the months leading up to the January comparative, risen quite sharply: this upward trend appears to be continuing, but perhaps losing a little momentum. The In the Wild score is identical, in percentage terms, to that of January; the product missed five boot sector viruses because it does not like to scan diskettes which DOS cannot access – it should be using lower level techniques to read the boot sector, which would avoid this problem. The In the Wild file virus missed was Alfons.1344.

If the product is to aspire to a higher score in the Polymorphic set, it must acquire further information on some of the more recent polymorphic viruses: the knowledge it currently possesses resulted in a score of only 59.7%.

NAV encountered a single false positive against the clean test-set. As far as the extra tests go, *NAV* detected all the boot viruses in memory, and removed them correctly from both types of disk. Both file viruses were detected when active, but only Manzon could be disinfected from files.

S&S Dr Solomon's AVTK v7.58

ItW Boot	100.0%	Standard	99.5%
ItW File	99.0%	Polymorphic	90.7%
ItW Overall	99.4%		

The *AVTK* misses out on 100% in the In the Wild sets because it failed to detect the five samples of Alfons.1344: this drops its score in this set to 99.4%; very close indeed. In the Polymorphic set, the gradual elimination of the incomplete detection problems that have previously troubled the product results in a score of 90.7%.

On the other tests, the *AVTK* found all the boot and file viruses in memory, and disinfected them from all infected objects without difficulty.

Trend Micro Devices PC-cillin v5.02.139

ItW Boot	97.4%	Standard	84.2%
ItW File	98.0%	Polymorphic	81.1%
ItW Overall	97.7%		

The scores obtained by this product against the In the Wild test-sets are slightly disappointing – they show that a little work needs to be done in this area before the product lives

up to its advertising. This should take little effort, however, as the score is far from irreconcilable. Standard and Polymorphic scores are also unexciting.

PC-cillin detected all the boot sector viruses in memory and disinfected them from hard and floppy disks with no difficulties. It missed both the file viruses when resident, but attempted disinfection.

Unfortunately, this product also has the dubious honour of being the only product in this test to 'break' executables in the course of disinfection: after disinfection, the 'cleaned' Burglar samples all produced 'Error in EXE file' messages. The disinfected Manzon samples had extra bytes left behind at the end of the file, but they functioned without problems. Finally, *PC-cillin* encountered two false positives.

A Conceptual Problem?

One of the viruses in the In the Wild test-set which caused problems was, predictably enough, Concept. It is interesting to note which products are not yet finding this, the most reported virus, and more specifically, why. *AVP*, *VET*, *DrWeb*, *PCVP*, *ViruSafe*, *VirusBuster*, *SWEEP*, and *PC-cillin* all omit the DOT and/or DOC extensions by default from their scan lists. Of these, *AVP*, *SWEEP*, and *PC-cillin* find all the Concept samples when explicitly asked to check the extensions in question.

The others still miss all or some of the samples – *PCVP* comes with a separate program with which to detect macro viruses (which it does successfully). This whole issue raises many problematical questions, some of which are addressed in this month's editorial (see p.2).

Conclusions

The *Virus Bulletin* comparative has, as always, provided the reader with an enormous amount of information, a summary of which has been shown on the preceding pages. Many conclusions may be drawn from these figures, dependent on how they are viewed and interpreted.

The scores on the In the Wild test-sets should be regarded as the most significant: it is these figures which describe how well the product works against those viruses which are a real threat to the user community. The *WildList* is the best available guide to this threat, and the list which was used in this comparative review was dated one month before the deadline for product submission – every product *should* be getting 100% against these sets. It is performance against the In the Wild test-sets which should influence any company's purchasing decisions.

The other test-sets are provided as an attempt to measure the performance of the scanner in various other areas – polymorphic detection is useful as a measure of the level of technology in the product. It is highly likely that a product which scores well here will have good core technology, at least in this one direction.

On a day-to-day basis, the average corporate should be very interested in scan time figures; these impact strongly on users, who should be scanning clean hard drives and disks on a daily basis. If a product takes too long to carry out these basic tasks, users will be unwilling to wait, and will stop using it. This is clearly undesirable – the perfect anti-virus product would be one which takes no time to run and finds all viruses.

The figures show that, as usual, *ESaSS ThunderBYTE* conquers all in the speed department – the time this product takes to scan an average hard drive will be fast enough to satisfy even the most critical user. *VET* and *PCVP* also scanned extremely quickly, but only managed to clock in at half the speed of *ThunderBYTE*.

Certain products are now using checksummers to help them speed the scanning process – *IBMAV* does this by default, and other products (notably *DrWeb*) come bundled with integrity checkers which will invoke the scanner on changed files. When no files have changed, the speed can be dramatically increased.

It is also very important to consider the impact of false positives; these can easily cause as much disruption as a genuine virus incident. There follows a breakdown of those products which suffered false positives:

Stiller Integrity Master	40
Leprechaun Virus Buster	18
DialogueScience DrWeb	12
ESaSS ThunderBYTE	5
RG Software Vi-Spy	5
H+BEDV AVScan	2
Trend PC-cillin	2
KAMI AVP	1
Symantec Norton AntiVirus	1

Conclusions

Taking all of the preceding information into account, the decision as to which product is 'best' is a tricky one. Given the importance of In the Wild detection, the two top products appear at first glance to be *Command Software F-Prot* and *ESaSS ThunderBYTE*.

Despite its speed advantage, *ThunderBYTE's* false positive rate is a little too high (i.e. above zero) for comfort. This leaves *F-Prot*, even with its low polymorphic detection rate, as the best counter in this review to the real-world threat.

In other areas, *AVP* gets the edge – its excellent polymorphic detection and virus removal are to be commended. Unfortunately, the near miss on the ItW sets, the false positive it encountered, and its slow speed all combine to make it impossible to recommend as a first line of defence: this product is best suited for use as a tool for administrators.

The recommendation for corporate workstations, therefore, is *Command Software F-Prot* for everyday use, backed up by *KAMI AVP* for emergencies.

TECHNICAL DETAILS

Hardware/Software: Main machine – *Compaq ProLinea 590*, 16MB RAM, 2.1GB disk, a 270MB *SyQuest* removable drive. Subsidiaries – Eight 386/486 workstations of varying configurations. Portables – Five *Amstrad* 386 portables; 1MB RAM, 20MB hard drives. All ran *MSDOS 6.22*.

Other technical information: All speed tests were conducted on the main machine described above. No network was involved for these tests, to keep the speed figures consistent. For the detection test, the main machine was configured as a *NetWare* server, and the eight subsidiaries used as clients. The March 1996 *WildList* was the source for the In the Wild test-sets.

WWW address for calculation information: http://www.virusbtn.com/Comparatives/Dos/199607/protocol.html

TEST-SETS

In the Wild Boot Sector Test-set. 77 samples (one of each of the following viruses):

15_Years, AntiCMOS.A, AntiCMOS.B, AntiEXE, Boot.437, BootEXE.451, Brasil, Bye, Chance.B, Chinese Fish, Crazy_Boot, Da_Boys, Diablo_Boot, Disk_Killer, DiskWasher.A, Empire.Int_10.B, Empire.Monkey.A, Empire.Monkey.B, EXEBug.A, EXEBug.C, EXEBug.Hooker, FAT Avenger, Feint, Finnish_Sprayer, Flame, Form.A, Form.C, Form.D, Frankenstein, J&M, Joshi.A, Jumper.A, Jumper.B, Junkie, Kampana.A, Leandro, MISiS, Mongolian Boot, MusicBug, Natas.4744, NYB, Parity_Boot.A, Parity_Boot.B, Peter, QRry, Quandary, Quox.A, Ripper, Russian_Flag, Sampo, Satria.A, She_Has, Stealth_Boot.B, Stealth_Boot.C, Stoned.16.A, Stoned.Angelina, Stoned.Azusa.A, Stoned.Bravo, Stoned.Bunny.A, Stoned.Daniela, Stoned.Dinamo, Stoned.June_4th.A, Stoned.Kiev, Stoned.LZR, Stoned.Manitoba, Stoned.Michelangelo.A, Stoned.No_Int.A, Stoned.NOP, Stoned.Standard, Stoned.Swedish_Disaster, Stoned.W-Boot.A, Swiss_Boot, Unashamed, Urkel, V-Sign, WelcomB, Wxyc.A.

In the Wild File Test-set. 312 samples, made up of:

_814 (3), Accept.3773 (5), Alfons.1344 (5), Anticad.4096.A (4), Anticad.4096.Mozart (4), Arianna.3375 (4), Avispa.D (2), Backformat.A (1), Bad_Sectors.3428 (5), Barrotes.1310.A (2), BootEXE.451 (1), Bosnia:TPE.1_4 (5), Burglar (3), Byway.A (1), Byway.B (1), Cascade.1701.A (3), Cascade.1704.A (3), Cascade.1704.D (3), Cawber (3), Changsa.A (5), Chaos.1241 (6), Chill (1), Concept (4), CPW.1527 (4), Dark_Avenger.1800.A (3), Datalock.920.A (3), DelWin.1759 (3), Die_Hard (2), Dir-II.A (1), DR&ET.1710 (3), Fairz (6), Fichv.2_1 (3), Finnish.357 (2), Flip.2153 (2), Flip.2343 (6), Freddy_Krueger (3), Frodo.Frodo.A (4), Ginger.2774 (2), Green_Caterpillar.1575.A (3), Helloween.1376.A (6), Hi.460 (3), Hidenowt (1), Jerusalem.1244 (6), Jerusalem.1808.Standard (2), Jerusalem.Sunday.A (2), Jerusalem.Zero_Time.Australian.A (3), Jos.1000 (3), Junkie (1), Kaos4 (6), Keypress.1232.A (2), Lemming.2160 (5), Liberty.2857.A (2), Little_Brother.307 (1), Little_Red.1465 (2), Macgyver.2803 (3), Maltese_Amoeba (3), Manzon (2), Markt.1533 (3), Mirea.1788 (2), Natas.4744 (5), Necros (2), Neuroquila (1), No_Frills.Dudley (2), No_Frills.No_Frills.843 (2), Nomenklatura (6), November_17th.800.A (2), November_17th.855.A (2), Npox.963.A (2), One_Half.3544 (5), Ontario.1024 (3), Pathogen:SMEG.0_1 (5), Ph33r.1332 (5), Phx.965 (3), Predator.2448 (2), Quicksilver.1376 (1), Sarampo (6), SatanBug.5000.A (2), Sayha (5), Screaming_Fist.II.696 (6), Sibylle (3), Sleep_Walker (3), SVC.3103.A (2), Tai-Pan.438 (3), Tai-Pan.666 (2), Tequila.A (3), Three_Tunes.1784 (6), Trakia.653 (1), Tremor.A (6), Trojector.1463 (6), Vacsina.TP-05.A (2), Vacsina.TP-16.A (1), Vampiro (2), Vienna.648.Reboot.A (1), Vinchuca (3), Virogen.Pinworm (6), VLamix (1), Xeram.1664 (4), Yankee Doodle.TP-39 (5), Yankee_Doodle.TP-44.A (1), Yankee_Doodle.XPEH.4928 (2).

Standard Test-set. 409 samples, made up of:

AIDS (1), AIDS-II (1), Alabama (1), Algerian.1400 (3), Amazon.500 (2), Ambulance (1), Amoeba (2), Annihilator.673 (2), Anston.1960 (5), Anthrax (1), AntiGus.1570 (3), Anti-Pascal (5), Argyle (1), Armagedon.1079.A (1), Assassin.4834 (3), Attention.A (1), Auspar.990 (3), Baba.356 (2), Backfont.905 (1), Barrotes.840 (3), Bebe.1004 (1), Big_Bang.346 (1), Billy.836 (3), Black_Monday.1055 (2), Blood (1), Blue_Nine.925.A (3), Burger (3), Burger.405.A (1), Butterfly.302.A (1), Cantando.857 (3), Cascade.1701.Jo-Jo.A (1), Casper (1), CeCe.1998 (6), CLI&HLT.1345 (6), Coffeeshop (2), Cosenza.3205 (2), Coyote.1103 (3), Crazy_Frog.1477 (3), Crazy_Lord.437 (2), Cruncher (2), Cybercide.2299 (3), Danish_Tiny.163.A (1), Danish_Tiny.333.A (1), Dark_Avenger.2100.A (2), Dark_Avenger.1449 (2), Dark_Revenge.1024 (3), Datacrime (2), Datacrime_II (2), DBF.1046 (2), Dei.1780 (4), Despair.633 (3), Destructor.A (1), Diamond.1024.B (1), Dir.691 (1), DOSHunter.483 (1), DotEater.A (1), Ear.405 (3), Eddie-2.651.A (3), Eight_tunes.1971.A (1), Fax_Free.1536.Topo.A (1), Fellowship (1), Fisher.1100 (1), Frodo.3584.A (2), Flash.688.A (1), Fumble.867.A (1), F-You.417.A (1), Genesis.226 (1), Green.1036 (6), Greets.3000 (3), Halloechen.2011.A (3), Hamme.1203 (6), Happy_New_Year.1600.A (1), HDZZ.566 (3), HLLC.Even_Beeper.A (1), HLLC.Halley (1), Horsa.1185 (3), Hymn.1865.A (2), Hymn.1962.A (2), Hymn.2144 (2), Hypervisor.3128 (5), Ibqqz.562 (3), Icelandic.848.A (1), Immortal.2185 (2), Internal.1381 (1), Invisible.2926 (2), Istanbul.1349 (6), Itavir.3443 (1), Jerusalem.Fu_Manchu.B (2), Jerusalem.PcVrsDs (4), Jerusalem.1808.CT.A (4), John.1962 (3), Joker (1), July_13th.1201 (1), June_16th.879 (1), Kamikaze (1), Kela.b.2018 (3), Kemerovo.257.A (1), Keypress.1280 (6), Kukac.488 (1), Leapfrog.A (1), Leda.820 (3), Lehigh.555.A (1), Liberty.2857.A (5), Liberty.2857.D (2), Loren.1387 (2), LoveChild.488 (1), Lutil.591 (3), Maresme.1062 (3), Metabolis.1173 (3), Mickie.1100 (3), Necropolis.1963.A (1), Nina.A (1), November_17th.768.A (2), NRLG.1038 (3), Omud.512 (1), On_64 (1), Oropax.A (1), Parity.A (1), Peanut (1), Perfume.765.A (1), Phantom1 (2), Phoenix.800 (1), Pitch.593 (1), Piter.A (2), Pixel.847.Hello (2), Pizelun (4), Plague.2647 (2), Poison.2436 (1), Pojer.4028 (2), Positron (2), Power_Pump.1 (1), Prudents.1205.A (1), Quark.A (1), Red_Diavolyata.830.A (1), Revenge.1127 (1), Riihi.132 (1), Rmc.1551 (4), Rogue.1208 (6), Saturday_14th.669.A (1), Screaming_Fist.927 (4), Screen+1.948.A (1), Semtex.1000.B (1), Senorita.885 (3), Shake.476.A (1), ShineAway.620 (3), SI.A (1), SillyCR.710 (3), Sofia.432 (3), Spanz.639 (2), Stardot.789.A (6), Stardot.789.D (2), Starship (2), Subliminal (1), Suomi.1008.A (1), Suriv_1.April_1st.A (1), Suriv_2.B (1), Surprise.1318 (1), SVC.1689.A (2), Svin.252 (3), Svir.512 (1), Sylvia.1332.A (1), SysLock.3551.H (2), TenBytes.1451.A (1), Terror.1085+A247 (1), Thanksgiving.1253 (1), The_Rat (1), Tiny.133 (1), Tiny.134 (1), Tiny.138 (1), Tiny.143 (1), Tiny.154 (1), Tiny.156 (1), Tiny.159 (1), Tiny.160 (1), Tiny.167 (1), Tiny.188 (1), Tiny.198 (1), Todor.1993 (2), Traceback.3066.A (2), TUQ.453 (1), Untimely.666 (3), V2Px.1260 (1), V2P6 (1), Vacsina.634 (1), Vacsina.700 (2), Vacsina.1212 (1), Vacsina.1269 (1), Vacsina.1753 (1), Vacsina.1760 (1), Vacsina.1805 (1), Vacsina.2568 (1), Vbasic.5120.A (1), Vcomm.637.A (2), VFSI (1), Victor (1), Vienna.Bua (3), Vienna.Monxla.A (1), Vienna.583.A (1), Vienna.623.A (1), Vienna.648.Lisbon.A (1), Vienna.W-13.507.B (1), Vienna.W-13.534.A (1), Vienna.W-13.600 (3), Virus-101 (1), Virus-90 (1), Voronezh.600.A (1), Voronezh.1600.A (2), VP (1), Warrior.1024 (1), Whale (1), Willow.1870 (1), WinVir (1), WW.217.A (1), Yankee_Doodle.1049 (1), Yankee_Doodle.2756 (1), Yankee_Doodle.2901 (1), Yankee_Doodle.2932 (1), Yankee_Doodle.2981 (1), Yankee_Doodle.2997 (1), Zherkov.1023.A (1), Zero_Bug.1536.A (1).

Polymorphic Test-set: 10,000 samples; 500 each of:

Alive.4000, Code.3952:VICE.05, Digital.3547, DSCE.Demo, Girafe:TPE, Gripe.1985, Groove and Coffee_Shop, MTZ.4510, Natas.4744, Neuroquila.A, Nightfall.4559.B, One_Half.3544, Pathogen:SMEG, PeaceKeeper.B, Russel.3072.A, SatanBug.5000.A, Sepultura:MtE-Small, SMEG_v0.3, Tequila.A, Uruguay.4.

PRODUCT REVIEW 1

Norton AntiVirus for NetWare

Martyn Perry

This month we review *Norton AntiVirus for NetWare* (NAVNLM) version 2 from *Symantec*. As with many other recently reviewed products, multi-server administration is supplied as an integral part of the product offering.

The product is licensed on a per-server basis, and supports both *NetWare 3.x* and *4.x*. When running with *NetWare 4.x*, NAVNLM supports *NetWare* Directory Services. The software for workstation protection requires separate licensing – workstation packs are available separately for DOS, *Windows*, *Windows 95* and *OS/2*.

Presentation and Installation

The product came with a manual and three diskettes. Two of the diskettes contained the server and administration components on *NAV*, whilst the third was a Virus Definitions update disk.

The documentation is comprehensive yet concise. It includes a useful glossary as well as a list of NLM and *Windows* error messages. There is a short description of the various types of virus, and a checklist for an emergency response plan.

Installation is performed from a workstation, under DOS or *Windows*. The default installation uses the subdirectory SYS:SYSTEM\NAVNLM, and C:\NAVNLM on the workstation – both can be user-defined. Another choice can be made during installation as to whether or not to modify AUTOEXEC.NCF (or MSAUTO.NCF on SFT III systems) to load NAVNLM automatically at server start-up.

If the automatic option is not chosen, the NAVNLM program is loaded from the server console prompt using the command Load [volume]:\[pathname]\nav.nlm. This loads the main NAV.NLM, plus two supporting NLMs.

Once loaded, the scanner can be controlled from the sever console by using NAVNLM with the various function keys. These keys allow the scanner to be enabled and disabled, an immediate scan to be started and stopped, and the NLM itself to be unloaded.

Each of the above operations require the *Novell* user name and password to be entered before being executed – a useful security measure.

Administration

Scanner administration is carried out using the *Windows* configuration program. The administration program can configure any server on which the NAVNLM is loaded. The software allows servers to be added to a NAVNLM domain, providing that the administrator has the necessary supervisor rights to those servers. Both *NetWare 3.x* (bindery) and *NetWare 4.x* (NDS) servers can be managed.

The main administration screen provides a view of the various servers and the protection status of the three types of scan. NAVNLM has the usual three modes of scanner operation: immediate, real-time and scheduled.

An immediate scan checks the server on demand, using the current immediate settings, and can be started either from the workstation or from the server. The real-time option allows scanning when a file is written to or read from the server. A scheduled scan provides checks on a timed basis – multiple scheduled scans are supported.

Configuration Options

For each mode of operation, selections can be made; first, of which file extensions are to be included in the scan. The default list is EXE, COM, OV?, SYS, BIN, APP, PRG, XTP, VXD, 386, DLL, DOC, DOT – extra file extensions can be added as necessary. The ability to define the item to be scanned down to a single file, if necessary, is a very useful facility.

In addition to file selection, items can be excluded either at the volume level or, again, all the way down to individual files. Another exclusion facility is available – this defines for which users the exclusion is intended (choose from everyone, supervisor, or named individuals), and the length of the exclusion (which can be forever, or until a certain time and date is reached). This option can be a useful facility for maintenance operations, providing that proper care is taken not to compromise the overall virus protection.

The options dialog on NAVNLM's administration utility offers great flexibility.

A separate menu option allows selection of various actions which can be taken on detection of a virus under any scan mode. The available choices are: deny access to the file, purge the infected file, rename the infected file with a user-defined extension, move the infected file to a user-defined quarantine directory, or leave the file alone.

Additional actions to perform on virus detection are available: an NLM may be loaded on the server, and the workstation that caused the alert may be logged off. There is one other option, which the product defines as 'scanning for unknown viruses'. More of this in a moment.

Alert Management

Apart from the action which can be taken when a virus is detected, the administration program can set up a list of users who will receive a network broadcast.

This list is defined as All Users, File User (the user accessing the file), File Owner, File Updater (the person who last modified the file), System Console (not a real user, but the message will be displayed on the server console), and Supervisor/Admin (users with supervisory access rights).

Alerts can also be sent to defined *Novell* groups, users with pagers, and users with MHS mail access. A dummy user can be set up to allow messages to be sent to other servers.

Inoculation

Now to consider how to deal with a new virus not known by the scanner. For each mode of operation, *Norton* provides the option to detect known and unknown viruses.

It transpires that 'detection of unknown viruses' is a euphemism for checksumming. Key areas of the file are fingerprinted: if they are altered in any way, the fingerprint will not match that stored. The change will act as an alert.

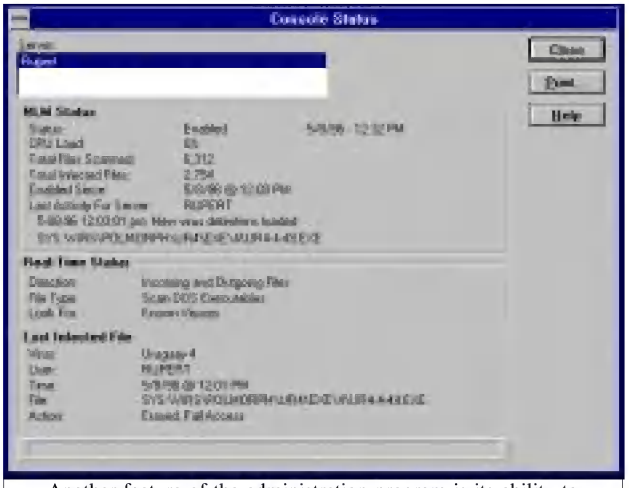
Reports and Activity Logs

NAVNLM keeps a record of events in an activity log. These events can be selected, and include:

- Detection of known and unknown viruses
- Scan start and end times
- Virus list changes
- Loading and unloading of NAVNLM
- Alerts from workstations
- Status and error messages

With this amount of data, control is necessary, and this is supplied in two ways. First, it is possible to limit the size of the log file and second, the events being displayed can be filtered. This filtering is further enhanced by the fact that date ranges and specific users may be selected as required.

Other reports include the current configuration, and the console status – this shows the information which is displayed on the console. All of these reports are available



Another feature of the administration program is its ability to 'seek out' remotely the data being displayed by the server.

for each server installed with NAVNLM, and the reports can be printed to hard copy, exported to a text file, or sent to another user.

Updates

The only problem which was encountered during the evaluation was when it came to updating the virus signatures. The documentation file supplied with the update disk describes how to update the various versions of the product. Under the other versions, mention is made of copying files from the SPECIAL directory. No mention of this directory is made for the NLM, although it is present on the update disk.

These files are not necessary for updating the NLM (apparently they are used only by older versions of the workstation products) – a fact which could be made clearer by an explicit mention in the documentation file. It would also be nice to have the update process automated in some way.

Detection Rates

The scanner was tested against the usual three test sets; In the Wild, Standard and Polymorphic (see summary for details). Undetected viruses were identified by using the 'purge files' option and listing the files left behind on the disk. The tests were conducted using the default scanner file extensions supplied [*covering everything in the test-set. Ed.*].

The run against the In the Wild test-set resulted in an excellent 99.3%, failing only on two samples of Virogen.Pinworm. There were several misses against the Standard set, but this result was still very creditable at 95.4% However, the run against the Polymorphic test-set yielded only 59.2% – this is particularly peculiar in view of *Norton's* marketing claim as 'Leader in polymorphic/Mutation Engine viruses'.

Real-time Scanning Overhead

To determine the impact of the scanner on the server when it is running, 63 files comprising a total of 4,641,722 bytes (EXE files from SYS:PUBLIC) were copied from one server

directory to another using *Novell's* NCOPY. The directories used for the source and target were excluded from the virus scan, to avoid the risk of a file being scanned while waiting to be copied.

As usual, because of the different processes which occur within the server, the time tests were run ten times for each setting, and an average was taken. A group of four tests were executed for two conditions.

The tests were run first without inoculation, to establish the effect of the scanner by itself on the server performance. They were then repeated with files inoculated, to see what impact the inoculation had on server performance. The four tests were:

- A/E: NLM not loaded. This establishes the baseline time for copying the files on the server.
- B/F: NLM loaded, using the default setting of on-access scanning for incoming and outgoing files, but the immediate scanner not running. This tests the impact of the on access (real-time) protection.
- C/G: NLM loaded, on-access scanning for incoming and outgoing files, and immediate scan running. This is the full worst-case impact of running the scanner.
- D/H: NLM unloaded. This is run to check how well the server is returned to its former state.

These tests were repeated to see the effect of using inoculation. See the summary for the detailed results.

The overhead of the on-access scanner on its own is quite low, but rises (as expected) when the immediate scanner is running. An option exists to control CPU usage which can mitigate this overhead for the immediate scanner.

NAVNLM periodically examines the server load during a background scan. By comparing the current server load with a target user defined load level, the NLM may adjust the delay that is inserted between files scanned.

The performance overhead of having the files checked under inoculation is only minimal; therefore, it is worthwhile considering this option as additional protection for the server. Since NAVNLM performs a clean unload of all three of the NLMs which were originally installed, the residual overhead is negligible.

Conclusion

The product is easy to install, and comes with good documentation. The scan options provide a wide range of choices for selection and exclusion, and the alerts are comprehensive and straightforward to configure.

An additional plus is the handling of the multi-server environment, which is a useful feature. The update facility, however, does need to be tidied up, so that the documentation matches the files on the update disk.

The detection rate, with the exception of the polymorphics, is very good. The critical In the Wild test-set is handled well, although not perfectly; it is really only on the polymorphics where the product falters. Overall, the features provided in the product make it one of the best I have reviewed so far, but more effort needs to be made with polymorphic detection to catch up with the marketing hype. [*Alternatively, reduce the hype... Ed.*]

Norton AntiVirus for NetWare

Detection Results

Test-set[1]	Viruses Detected	Score
In the Wild	298/300	99.3%
Standard	291/305	95.4%
Polymorphic	4734/8000	59.2%

Overhead of On-access Scanning:

Tests detail the time taken to copy 63 EXE files (4.6MB); average time in seconds for 10 tests.

	Time	Overhead
Known viruses only		
A. NLM not loaded	10.9	n/a
B. NLM loaded, no manual scan	12.6	15.6%
C. NLM loaded, manual scan	36.3	233.0%
D. NLM unloaded	10.9	n/a
Known and unknown viruses		
E. NLM not loaded	10.9	n/a
F. NLM loaded, no manual scan	12.7	16.5%
G. NLM loaded, manual scan	36.4	233.9%
H. NLM unloaded	10.9	n/a

Technical Details

Product: *Norton AntiVirus for NetWare.*

Developer/Vendor: *Symantec Corporation*, 10201 Torre Avenue, Cupertino, California 95014, USA. Tel +1 408 253 9600, fax +1 408 453 0636, BBS +1 503 484 6669, WWW: http://www.symantec.com/.

UK Distributor: *Symantec UK*, Sygnus Court, Market Street, Maidenhead, Berkshire SL6 8AD. Tel 01628 592222, fax 01628 592393.

Price: US pricing – single server licence (including 10 workstations) US$599.00.
UK pricing – 1 x NAVNLM protecting one server only £597.00. Ten additional users for workstations @ £99.00 each.
Site Licence prices available on request.

Hardware Used:
Server – *Compaq Prolinea 590* with 16MB Ram and 2 GB of hard disk, running *NetWare 3.12.*
Workstation – *Compaq 386/20e* with 4MB Ram and a 207 MB hard disk, running *DOS 6.22* and *Windows 3.1.*

[1]**Test-sets:** For a complete listing of all the viruses used in these tests, please refer to *Virus Bulletin*, January 1996, p.20.

PRODUCT REVIEW 2

On the FrontLine

Dr Keith Jackson

FrontLine is a package that provides protection against boot sector viruses, and claims to 'guarantee to detect all boot viruses on diskette'. It alters the hard disk partition sector, and uses a memory-resident program to detect/eradicate viruses: it has no knowledge of any sort of virus other than the boot sector type. The developers state that it will find parasitic viruses which go resident just below the 640KB memory ceiling: this is, of course, not a complete solution.

The product works under DOS, *Windows 3.1* or *Windows 95*. Parts of *FrontLine* claim to be network aware: as I have no means of testing *Windows 95* software or network-enabled features, neither of these parts will be assessed or discussed.

FrontLine claims to be 'the only known anti-virus program in this world that can detect 100% of all boot viruses'. I cannot test the implied claim that a package in another world may do this, but even in this world there are similar products which only offer detection/removal of boot sector viruses.

Last year I reviewed a product called *No.More #*!$ Viruses* [*VB, November 1995, p.21*]. Readers will notice the similarity between that product and *FrontLine*, although *FrontLine* uses memory-resident software to check floppy disks as they are used, which *No.More* does not.

Method of Operation

FrontLine stores within itself signatures of all the legal boot sectors which may be found in a PC, from *MS-DOS* (v2.2 to 6.22), other species of DOS, *Windows 95*, *Windows NT*, OS/2, some types of Unix, and *Mac* boot sectors. It claims to know 'other proprietary boot sectors', but information as to what these are is not provided. The extant boot sector is compared to the known valid boot sectors – changes are termed an 'unregistered boot sector', or (if a known virus is spotted) a 'suspected virus'. In reality, both errors occur (see results).

FrontLine claims 'The false alarm rate is almost nil'. Hmmm. Almost? I would be happier had figures been provided to back this up. Software that produces any non-trivial level of false alarms is close-on useless, even counter-productive, as the alarms soon get ignored. Although I experienced no false alarms whilst using *FrontLine*, this is always likely to be true on what is necessarily a very small sample size.

Documentation

The documentation comprised a few explanatory sheets of A4, and an executable file described as 'electronic documentation'. The latter proved far more than text wrapped inside an executable file: it contained the manual for *FrontLine* in the form of a *Word Perfect Envoy* viewer package. This provides word processor features associated with viewing/printing files in a format similar to *Word Perfect for Windows*.

My only gripe about this format is the time it took to load. On the 486 PC used for this review it took almost 30 seconds before I was able to view the documentation, and on the PC used for testing, loading took a whopping 1 minute 42 seconds. I contend that this makes this style of documentation unusable on many of the older, slower PCs still in use.

The documentation is a bit rambling, but it does explain the concept behind *FrontLine* quite clearly, and gives plenty of helpful information for those installing *FrontLine* for the first time. This is really a 'fit and forget' product, so I suppose there is no need for voluminous explanatory books.

Installation

FrontLine was supplied for review on a 3.5-inch (1.44 MB) floppy disk. Installation involved executing a batch file, which makes a subdirectory called C:\FRNTLINE, copies files there, adds a line to the start of AUTOEXEC.BAT, and executes a program which alters the partition table of the hard disk. The name of the subdirectory into which the *FrontLine* files are installed cannot be changed. This should be rectified. It's my hard disk: I want to put files where I choose, not where a product tells me to put them.

During installation, the system is checked for viruses, but it does not state in detail how the checks are carried out. For instance, is a check made for multi-partite viruses? A reboot is performed, so the system can be booted from the altered hard disk partition. Installation is very quick, taking just 19 seconds until the aforementioned system reboot occurs.

After *FrontLine* has been installed, the memory-resident program occupies a reasonably frugal 7 KB of memory. The only interrupt captured by this component is Int 13h, which in my reference books is denoted as 'BIOS floppy disk services'. Only floppy disk requests are captured.

Routine Operation

After installation, *FrontLine's* memory-resident program is executed when the system is rebooted: this checks for boot sector viruses under DOS or *Windows*. If a boot sector virus is detected, program execution is temporarily halted, a warning message is displayed, and the user is invited to confirm that the virus should be removed immediately.

If *Windows* is in use, such messages can only be displayed if a special *Windows*-aware program has previously been executed. This component is copied to the hard disk by the installation program, but the user must create a *Windows* icon for the program to be made executable under *Windows*.

If the *Windows* component has not been executed, *FrontLine* merely gives a beep whenever a boot sector virus is detected. This is somewhat limited in its impact on users.

Detection of Virus Infection

I tested the product's virus detection capability against the boot sector viruses in the test-set described in the Technical Details section. For each of the twenty boot sector samples, *FrontLine* correctly, and unsurprisingly, detected a virus infection. However, it did not know which virus was present, it merely reported that something was awry.

In all cases, *FrontLine* either stated 'Suspected Boot Virus detected', or 'Unregistered boot sector'. This was always followed by instructions to 'notify the System Administrator'. If an 'Unregistered boot sector' is found, the developers ask that a sample is sent to them for further analysis – a free software upgrade is offered to encourage this process.

Of the twenty test samples used, *FrontLine* detected sixteen as 'suspected viruses', and four (Da'Boys, Junkie, Ripper and Urkel) as 'Unregistered boot sectors'. This means that, although *FrontLine* knew what the sixteen suspected viruses were, with the remaining four it knew only that the boot sector was not as it should be, and not which virus was involved – hence the request for a sample to be sent to the developers. DOS and *Windows* detection was identical.

The user is also asked to confirm whether *FrontLine* should immediately remove the virus. When the product's *Windows* component is used, a Yes/No answer is required, but pressing the 'Y' key to indicate assent does not work. If the virus is not removed immediately, *FrontLine's* messages may appear more than once, depending on how often the boot sector is accessed.

The results show that *FrontLine* performs its claimed task very well. It really does detect *any* boot sector virus infection… even if it doesn't always know what type of infection has occurred.

Removal and Clashes

FrontLine can successfully remove a boot sector virus infection, but given that this just means writing a non-infected boot sector to the disk in question, this is not very difficult to achieve. Indeed there does not seem to be a lot of difference between *FrontLine's* boot sector virus removal and the DOS FDISK program with its hidden /MBR switch (which implants a known good boot sector on the hard disk).

FrontLine is 'working fine' [*sic*] in memory.

As *FrontLine* changes the hard disk partition sector, it is almost inevitable that it will clash with other packages which use the same tactic. For instance, when a version of DOS is upgraded, it is necessary first to uninstall *FrontLine*, and reinstall it after the new DOS has been installed.

The four main problem areas with using this product are listed in the documentation as security access control software, IDE disks above 540 MB, hardcards (much rarer than they used to be), old XTs and ATs (which needed the partition sector at a different location). Of these, only the first two are likely to be a problem. The documentation describes how to proceed if either problem is encountered.

Conclusions

FrontLine prevents boot sector virus transmission by spotting the floppy where the infection originated, and preventing it spreading. It achieves this task with admirable precision. Indeed it seems to recognise *any* boot sector alteration.

As *FrontLine* changes the partition sector of the hard disk, it gets a chance to become operational before DOS has loaded. This is to be applauded in security terms, but it must be recognised that many other types of program also change the partition sector, and problems/clashes are almost inevitable.

As I pointed out in the review of *No.More #*!$ Viruses*, a problem lurks behind this seemingly limitless capability. This type of product does not address the problem of file-infectors at all, yet *FrontLine* installation should 'only be done on a virus-free system'. Although the matter is not discussed in depth in the documentation, *FrontLine* provides no thorough means of ensuring this is true for non-boot-sector viruses.

It is ironic that boot sector infections have been the most prevalent infections for many years, and at the very moment that products specific to their detection appear, macro viruses appear and take over the virus 'league table'. Such is life.

Technical Details

Product: *FrontLine*, version 2.93, serial number FLI-059.

Developer/Vendor: *Hiwire Computer & Security Pte Ltd*, 1104A Serangoon Road, Singapore 328197. Tel +65 3852040, fax +65 2341793, WWW http://home.pacific.net.sg/~hiwire.

Availability: DOS, *Windows 3.1* or *Windows 95*. The product needs 7 KB of available RAM.

Price: In Singapore dollars. Single user $69.90; 2-10 users $299; site licences on request. Updates twice yearly. Subscription renewal $20 per single PC, or 20% of site licence price.

Hardware used: A *Toshiba 3100SX*; a 16 MHz 386 laptop with a 3.5-inch (1.4 MB) floppy disk drive, a 40 MB hard disk and 5 MB RAM, running under *MS-DOS v5.00* and *Windows v3.1*.

Viruses used for testing purposes: The test-set contains 20 boot sector viruses, one each of AntiCMOS.A, AntiEXE, Da_Boys, Empire.Monkey.B, EXE_Bug.A, Form.A, IntAA, Jumper.B, Junkie, Natas.4744, NYB, Parity_Boot.B, Peanut, Quox, Ripper, Sampo, She_Has, Stoned.Angelina, Unashamed, Urkel.

SUBSCRIPTION RATES

Subscription price for 1 year (12 issues) including first-class/airmail delivery:

UK £195, Europe £225, International £245 (US$395)

Editorial enquiries, subscription enquiries, orders and payments:

Virus Bulletin Ltd, 21 The Quadrant, Abingdon, Oxfordshire, OX14 3YS, England

Tel 01235 555139, International Tel +44 1235 555139
Fax 01235 531889, International Fax +44 1235 531889
Email editorial@virusbtn.com
CompuServe address: 100070,1340
World Wide Web: http://www.virusbtn.com/

US subscriptions only:

June Jordan, *Virus Bulletin*, 590 Danbury Road, Ridgefield, CT 06877, USA

Tel +1 203 431 8720, fax +1 203 431 8165

END NOTES AND NEWS

S&S International is presenting **Live Virus Workshops** at the *Hilton National* in Milton Keynes, Buckinghamshire, UK on 2/3 September and 7/8 October 1996. Details are available from the company: Tel +44 1296 318700, fax +44 1296 318777.

***Sophos Plc's* next anti-virus workshops** will be on 24/25 July and 25/26 September 1996 at the training suite in Abingdon, UK. The two-day seminar costs £595 + VAT. One single day may be attended at a cost of £325 + VAT (day one: Introduction to Computer Viruses; day two: Advanced Computer Viruses). For information, contact Julia Line, Tel +44 1235 544028, fax +44 1235 559935, or access the company World Wide Web page (http://www.sophos.com/).

Several anti-virus software companies have reported the presence of a new virus in the wild. **Hare.7610 (also known as Krsna)** has been reported in Australia, Europe, and North America. Although it has some difficulty in replicating, it has still managed to spread successfully – probably, according to one vendor, due to the Internet. Look for an analysis in next month's *VB*.

Reflex Magnetics has several courses coming up: **Live Virus Experiences** (9/10 October), The Hacking Threat (24-26 July), and Internet Security and Firewalls (22 July). Further information is available from Rae Sutton: Tel +44 171 372 6666, fax +44 171 372 2507.

McAfee Associates has announced the launch of *WebShield*, 'the industry's first secure **anti-virus software solution for network firewalls and Internet gateways**', as recorded in a recent press release. Information on this and other *McAfee* products from the company; Tel +1 408 988 3832; in the UK, Tel 01344 304730.

Readers are reminded that the *6th Annual Virus Bulletin Conference and Exhibition* takes place in **Brighton, UK, on 19/20 September 1996**. Contact Alie Hothersall, Tel +44 1235 555139, for details.

Virus Bulletin is still receiving reports of infections resulting from use of a copy of the screensaver 'DogZone' (also known as 'Dogz'), which was made available on the Internet. **Some copies are infected with the *Windows* virus Tentacle**.

The *NCSA* is hosting the ***Web, Internet Security and Firewall Conference***, which will be held in San José, California from 30 September to 1 October. Details on the event can be obtained from the *NCSA*; Tel +1 717 258 1816, fax +1 717 243 8642, or email fwcon96west@ncsa.com. Information also available from their WWW site: http://www.ncsa.com/fw96west.html.

The *Computer Security Institute* (*CSI*) *23rd Annual Computer Security Conference* is to be held from 11 to 13 November in Chicago, Illinois, USA. The event will feature a program of over 120 sessions, including presentations on **Internet security, access, email**, etc. It will also include an exhibition of computer security products – free passes to attend the exhibit available from the *CSI*. Details on attending from the *CSI*; contact Patrice Rapalus on Tel +1 415 905 2310; email prapalus@mfi.com.

Yet **another virus solution designed to protect email gateways** has been released in the UK: *NetConnect's* offering, *EVE* (*Email Virus Eliminator*), claims to detect viruses in email attachments. For further information, contact Rob White and *NetConnect*; Tel +44 1223 423523, or email rwhite@netconnect.co.uk.

Finally, just in from the **Misguided Technical Reviews desk:** *PC World Magazine*, July 1996, discussing OS/2: 'The memory handling is via the High Performance File System (HPFS). It doesn't have the cluster size penalties of FAT, so there is no more jiggling between extended and expanded memory and I gained disk space on a larger drive.' Well, that's cleared that up.